NETWORK ALGORITHMS FOR CONTROL AND COMMUNICATION FOR IOT APPLICATIONS

A Dissertation

by

PING-CHUN HSIEH

Submitted to the Office of Graduate and Professional Studies of
Texas A&M University
in partial fulfillment of the requirements for the degree of

DOCTOR OF PHILOSOPHY

Chair of Committee,	I-Hong Hou
Committee Members,	P. R. Kumar
	Jean-Francois Chamberland
	Natarajan Gautam
Head of Department,	Miroslav Begovic

August 2018

Major Subject: Electrical Engineering

ABSTRACT

Internet of Things (IoT) technology is accelerating the integration of wireless communication networks and physical systems, such as the emerging networked transportation systems and industrial automation applications. By leveraging IoT, these physical systems are envisioned to perform critical tasks more reliably and efficiently based on the real-time control information provided by various types of sensors. To achieve this vision, there are many fundamental challenges to be tackled in both theory and implementation for control and communication of these physical systems. Specifically, we study the design and implementation of network algorithms for the following IoT applications:

- We develop scheduling schemes for networked transportation systems. Different from the conventional scheduling problem in computer networks, we consider practical constraints of the physical systems, such as switch-over delay, estimation errors, finite buffer sizes, and partially-connected systems and propose a throughput-optimal scheduling policy.

- We develop wireless network algorithms for collecting and disseminating critical control information. We design distributed algorithms for real-time wireless ad hoc networks. Moreover, we design scheduling algorithms for optimizing Quality of Experience for video delivery applications by applying diffusion approximation.

- We develop a low-latency wireless testbed for prototyping real-time wireless scheduling policies as well as the proposed network algorithms.

DEDICATION

To my mom and dad, and my beloved wife Yu Lin

ACKNOWLEDGMENTS

First and foremost, I would like to express my deepest gratitude to my advisor, Professor I-Hong Hou, for his guidance and support throughout my PhD study. I still remember clearly the very first meeting with him back in September 2014. During that meeting, I was so amazed by the creativity, elegant formulation and rigorous theory of his dissertation work on real-time wireless networks. Since then, I have been continuously inspired by his intuition, creativity, and insights into the heart of research problems. To me, he is an ideal advisor, who is always willing to sit down and brainstorm with me, provide constructive advice, and give me sufficient freedom in exploring new research problems. From him, I have come to realize that doing research is the most joyful mental grind in the world.

I would like to give special thanks to Professor P. R. Kumar, who is my role model in both research and life. In the first semester of my PhD study, I attended a talk given by Prof. Kumar on "Some thoughts on graduate studies, research, and life thereafter." This talk turned out to be the most important seminar that I have ever attended in my graduate studies. I was deeply inspired by his profound advice about research: "Start with a practical problem, and try to get to the heart of it. The real world is very rich and admits a lot of new ideas." His advice gave me so much confidence in exploring the unknown, especially during my period of wilderness.

I would like to acknowledge Professors Jean-Francois Chamberland and Natarajan Gautam for serving on my dissertation committee. Their inspiring suggestions have significantly improved the quality of this dissertation.

I also had a great fortune to collaborate with Professor Srinivas Shakkottai, who is a great mentor as well as a good friend to me. I benefited a lot from the weekly

discussions with him on the software-defined wireless testbed.

I would also like to thank Professor Eytan Modiano at MIT. My short visit to MIT in June 2017 was an invaluable experience. His advice and encouragement gave me a great deal of confidence in pursuing an academic career after graduation.

Special thanks to my colleague and roommate, Xi Liu, for introducing me to the research problems of networked transportation systems and discussing interesting research problems in our day-to-day conversations. I would like to thank my collaborators: Simon Yau for introducing me to the software-defined wireless testbed, and Jian Jiao for educating me with the simulators for transportation research.

I also want to thank my friends and colleagues: Charles Tsai, Meng-Jie Hsiao, Tao Zhao, Han Deng, Victor Lin, Yue Yang, Shuai Zuo, Jin Xu, Mohammadhussein Rafieisakhaei, and Bharadwaj Satchidanandan, for their company and encouragement.

CONTRIBUTORS AND FUNDING SOURCES

Contributors

This work was supported by a dissertation committee consisting of Professor I-Hong Hou (academic advisor) and Professors P. R. Kumar and Jean-Francois Chamberland of the Department of Electrical and Computer Engineering and Professor Natarajan Gautam of the Department of Industrial and Systems Engineering.

The simulations depicted in Chapter 2 were conducted in part by Jian Jiao of the Department of Civil Engineering. The experiments depicted in Chapter 5 were conducted in part by Simon Yau, Rajarshi Bhattacharyya, and Kartic Bhargav K. R. of the Department of Electrical and Computer Engineering.

All other work conducted for the dissertation was completed by the student independently.

Funding Sources

Graduate study was supported in part by NSF under contract number CNS-1719384, the U.S. Army Research Laboratory and the U.S. Army Research Office under contract/Grant Number W911NF-15-1-0279, Office of Naval Research under Contract N00014-18-1-2048, and NPRP Grant 8-1531-2- 651 of Qatar National Research Fund (a member of Qatar Foundation).

TABLE OF CONTENTS

Page

LIST OF FIGURES

FIGURE Page

LIST OF TABLES

TABLE Page

1. INTRODUCTION

Internet of Things (IoT) technology is accelerating the convergence of wireless communication networks and physical systems. For example, in networked transportation systems, vehicles are equipped with various sensors and wireless communication capability, and therefore are able to exchange critical information with the infrastructure and other neighboring vehicles. In industrial automation applications, a large amount of sensors and actuators are deployed in one manufacturing area to enforce critical manufacturing tasks in a timely manner. With the real-time information provided by IoT, these physical systems are envisioned to perform much more reliably and efficiently.

To realize this vision, there are still many fundamental challenges to be met regarding the control and communication for these IoT-assisted physical systems. In this dissertation, we are particularly interested in the following challenges:

- **(C1) With the collected real-time sensor information, how can we apply these data to control physical systems more efficiently?**
- **(C2) How to design network algorithms to reliably and efficiently collect and disseminate sensor data, either real-time or non-real-time, through wireless?**
- **(C3) How to prototype these network algorithms to achieve the wireless capacibility required by many IoT applications?**

To address (C1), the proposed research focuses on intelligent traffic control for networked transportation systems in IoT scenarios. Our goal is to design intelligent scheduling algorithms for signalized intersections based on real-time traffic informa-

tion, such as queue length and waiting time. The details of this research are described in Chapter 2.

For (C2), we propose to design distributed wireless protocols to improve timely-throughput performance for real-time wireless ad hoc networks. Meanwhile, we study scheduling algorithms for video delivery to optimize quality of experience (QoE). The details of this research are discussed in Chapter 3 and 4.

For (C3), we propose to develop a low-latency software-defined wireless testbed for rapid prototyping of network algorithms. To demonstrate the low-latency capability of our testbed, we will implement the proposed network algorithms as well as other real-time wireless scheduling algorithms on this testbed. The details of this research are illustrated in Chapter 5.

1.1 Main Contributions and Outline of the Dissertation

1.1.1 Throughput-Optimal Scheduling for Multi-Hop Networked Transportation Systems

In Chapter 2, we study the scheduling problem for signalized intersections in networked transportation systems. With the emerging IoT technology, vehicles and road infrastructure are connected via wireless and allowed to exchange information in real time. By leveraging IoT for transportation systems, we can draw an analogy between transportation systems and communication networks: a vehicle corresponds to a packet, an intersection correspond to a router, and a lane correspond to a queue. Moreover, duing the transition from the green to red phase, a guard time of 3 to 8 seconds is required to ensure safety. The throughput during this transition phase is nearly zero. The above problem can be formulated as a scheduling design problem of multi-hop queueing networks with switch-over delay. We propose an online scheduling policy which achieves throughput optimality with switch-over delay.

The proposed policy is completely decentralized in the sense that each intersection requires only local queue information. The proposed policy remains throughput-optimal when there are both connected and fixed-time intersections. With IoT technology, the proposed policy can be easily incorporated into the current transportation systems without additional infrastructure. Through extensive simulation in VISSIM, we show that our policy indeed outperforms the existing popular policies.

1.1.2 QoE-Optimal Scheduling for Wireless Video Delivery

In Chapter 3, we turn our attention to the scheduling problem of a network of multiple wireless video streams with an aim of optimizing Quality of Experience (QoE). The QoE of each flow is measured the duration of playback interruption experienced by the corresponding video user. We consider wireless systems where an access point (AP) transmits video content to clients over fading channels. We proposes online scheduling policies to optimize quality of experience (QoE) for video-on-demand applications in wireless networks. We are particularly interested in systems operating in the heavy-traffic regime. We first consider a special case of ON-OFF channels plus constant-bit-rate videos and establish a scheduling policy that achieves every point in the capacity region under heavy-traffic conditions. This policy is then extended for more general fading channels and variable-bit-rate videos, and we prove that it remains optimal under some mild conditions. We then formulate a network utility maximization problem based on the QoE of each flow. We demonstrate that our policies achieve the optimal overall utility when their parameters are chosen properly. Finally, we compare our policies against three popular policies. We provide entensive simulation and experimental results to validate that the proposed policies indeed outperform existing policies.

1.1.3 Decentralized Medium Access Protocols for Real-Time Wireless Ad Hoc Networks

In Chapter 4, we consider the problem of optimizing medium access for real-time ad hoc networks, where a strict deadline is imposed for each packet. We are particularly interested in real-time wireless networks for industrial IoT applications, where multiple access points serve a network of sensors and actuators with the need of exchanging time-critical information. While centralized scheduling algorithms provide provably optimal theoretical guarantees, they may not be practical in such settings since it can be extremely difficult to achieve coordination among APs with stringent per-packet deadlines. Therefore, it is of great importance to design a wireless protocol that achieves feasibility optimality in a decentralized manner. To design a decentralized protocol, we leverage two widely-used functions of wireless devices: carrier sensing and backoff timers. Different from the conventional approach, the proposed protocol utilizes a collision-free backoff scheme to enforce the transmission priority of different links. This design obviates the capacity loss due to collision with quantifiably small backoff overhead. The algorithm is fully decentralized in the sense that every link only needs to know its own priority, and links contend for priorities only through carrier sensing. We prove that the proposed algorithm is feasibility-optimal. NS-3 simulation results show that the proposed algorithm indeed performs as well as the feasibility-optimal centralized algorithm.

1.1.4 A Low-Latency Software-Defined Wireless Testbed for Prototyping Wireless Protocols

In Chapter 5, we introduce a low-latency software-defined wireless testbed for prototyping wireless medium access control (MAC) protocols. An increasing number of emerging applications, such as virtual reality (VR) and tactile Internet, require

packets to arrive before a certain deadline for the system to have the desired performance. While many real-time wireless scheduling protocols have been proposed, few have been experimentally evaluated to establish realistic performance. Furthermore, some of these protocols involve high complexity algorithms that need to be performed on a per-packet basis. Experimental evaluation of these protocols requires a flexible platform that is readily capable of implementing and experimenting with these protocols. We present PULS, a processor-supported ultra lowlatency scheduling implementation for testing of downlink scheduling protocols with ultra-low latency requirements. Based on our decoupling architecture, programmability of delay sensitive scheduling protocols is done on a host machine, with low latency mechanisms being deployed on hardware. This enables flexible scheduling policies on software and high hardware function re-usability, while meeting the timing requirements of a MAC. We performed extensive tests on the platform to verify the latencies experienced for per-packet scheduling, and present results that show packets can be scheduled and transmitted under 1 ms in PULS. Using PULS, we implemented four different scheduling policies and provide detailed performance comparisons under various traffic loads and real-time requirements. We show that in certain scenarios, the optimal policy can maintain a loss ratio of less than 1% for packets with deadlines, while other protocols protocols experience loss ratios of up to 65%.

Finally, we conclude the dissertation by providing the concluding remarks and promising future research in Chapter 6. For better readability, some of the proofs are provided in the Appendices.

2. THROUGHPUT-OPTIMAL SCHEDULING FOR MULTI-HOP NETWORKED TRANSPORTATION SYSTEMS[1]

2.1 Overview

In this chapter, we study the scheduling problem for traffic control of signalized intersections for networked transportation systems. Recently, there have been more and more research works on exploring novel scheduling strategies for intersections from the perspective of networked transportation systems, which incorporate emerging IoT technologies such as vehicle-to-vehicle (V2V) communication and vehicle-to-infrastructure (V2I) communication. By leveraging IoT technologies, roadside infrastructure can obtain accurate traffic information in real time, such as the number of vehicles waiting in each lane [2, 3]. The scheduling problem in networked transportation systems then becomes very similar to that in computer networks. Specifically, we can draw the following analogy: each intersection corresponds to a router, each lane corresponds to a queue, and each vehicle corresponds to a packet. In this analogy, there indeed have been efforts to apply the queue-length-based Max-Pressure scheduling policy of computer networks [4] to networked transportation systems [5, 6, 7, 8].

While these attempts are promising, it might not be practical to direclty apply these scheduling algorithms to networked transportation systems. Currently, most scheduling algorithms manage traffic flows at intersections via traffic signals, which switch periodically between red and green. Transition from the green to red phase is not instantaneous, but requires a guard time for safety, usually of about 3-8 sec-

[1]Reprinted with permission from "Throughput-Optimal Scheduling for Multi-Hop Networked Transportation Systems With Switch-Over Delay" by Ping-Chun Hsieh, Xi Liu, Jian Jiao, I-Hong Hou, Yunlong Zhang, and P. R. Kumar in Proc. of ACM MobiHoc 2017 [1].

onds [9]. The throughput during this transition phase is nearly zero. In addition, there is also throughput loss when a new green phase starts or ends because of acceleration or deceleration of vehicles. We capture such capacity loss by introducing *switch-over delay* in this research. The switch-over delay needs to be explicitly addressed in designing scheduling policies for intersections. Unfortunately, most of the existing literature on scheduling intersections via traffic signals ignores the effect of switch-over delay. In fact, Ghavami *et al.* [10] demonstrate that, while dynamic signal control policies such as the Max-Pressure policy outperforms conventional fixed-time policies in general, the performance of the dynamic signal control policies can be seriously affected by capacity loss when switch-over delay is considered.

Furthermore, during the transition from a traditional transportation system to a fully connected system, only some of the intersections are equipped with sensors and V2I/V2V communication [11], while the rest relying on conventional fixed-time control policies. In such *partially-connected systems*, any new proposed policies will need to coexist well with conventional ones.

Based on the above discussion, our goal is to design a scheduling algorithm that addresses all of the following challenges:

- The scheduling algorithm is required to achieve throughput-optimality under switch-over delay.
- The scheduling algorithm does not require any knowledge of traffic demands.
- The algorithm needs to be distributed and scales well with network size.
- The algorithm shall be robust to error in the sensor data, i.e. preserve throughput-optimality with estimation error.
- The algorithm is able to coexist with the conventional fixed-time ones in

partially-connected systems.

2.2 Related Work

In current transportation systems, traffic signals are often adaptively controlled by proprietary traffic control suites, such as SCATS [12] and SCOOT [13]. Following the fixed-time control paradigm, these software suites require real-time traffic statistics to optimize cycle splits and offsets in the timing plan for given objective functions. However, traffic demand can change rapidly with time, and it is difficult and costly to collect the required statistics in a timely manner.

Differing from the fixed-time approach, scheduling design based on real-time queue length information is attracting increasing attention due to recent progress in connected-vehicle technology. For example, adaptive control based on queue length is proposed in [2], where the queue length is estimated via probe vehicles with V2I and V2V communication. On the other hand, inspired by results in computer networks [4], Varaiya [5] and Wongpiromsarn *et al.* [6] propose a Max-Pressure policy for signal control and formally prove that it is throughput-optimal when the queue capacity is infinite and the routing rates are known. To relax the assumption of infinite queue capacity, Xiao *et al.* [7] present a variation of the Max-Pressure policy that is throughput-optimal within a reduced capacity region when the queue capacity is finite but large enough. To relax the assumption on routing rates, Gregoire *et al.* [8] also propose a back-pressure-based signal control policy and prove that it is throughput-optimal with unknown routing rates. Despite the above progress, none of these policies takes the switch-over delay into account.

In the existing literature on the scheduling design for systems with switch-over delay, [14, 15, 16, 17] are the most relevant to the scope of this research. Armony and Bambos [14] study a system of parallel queues with switch-over delay and propose

a family of dynamic cone policies and batch policies to achieve optimal throughput. Subsequently, Hung and Chang [15] present a generalized version of the dynamic cone policy to reduce the complexity of the original cone policy. Chan [16] also presents a Max-Weight type policy with hysteresis and prove that it is throughput-optimal for a system of parallel queues with deterministic service processes. Celik *et al.* [17] propose a family of generalized Max-Weight policies and prove that any policy satisfying the proposed criteria is throughput-optimal. As an example in [17], the Variable Frame-Based Max-Weight (VFMW) policy introduces a frame structure to avoid excessive capacity loss due to switch-over delay. However, all the above policies are designed specifically only for single-hop systems and hence the optimality results may not carry over multi-hop systems. In this research, we regard VFMW as the reference policy for comparison in the simulations. In Section 2.6.4, we show that the VFMW policy, which is throughput-optimal for single-hop systems, can actually perform poorly in multi-hop systems.

2.3 System Model

2.3.1 Network Topology

We model a multi-hop transportation system by a directed graph $(\mathcal{V}, \mathcal{L})$, where $\mathcal{V}$ denotes the set of intersections and $\mathcal{L}$ is the set of directional links connecting the intersections. Each link has a start node and an end node. In this chapter, we use the terms *node* and *intersection* interchangeably. For convenience, we also include a common virtual source node v_{s} as well as a common virtual destination node v_{d} in the directed graph. We assume time is slotted. The links can be further divided into three categories: internal links $\mathcal{L}_{\mathrm{int}}$, entry links $\mathcal{L}_{\mathrm{entry}}$, and exit links $\mathcal{L}_{\mathrm{exit}}$. Each entry link has the same start node v_{s} and an end node $v \in \mathcal{V}$ where $v \neq v_{\mathrm{d}}$. Similarly, each exit link has the same end node v_{d} and a start node $v \in \mathcal{V}$ where $v \neq v_{\mathrm{s}}$. Therefore,

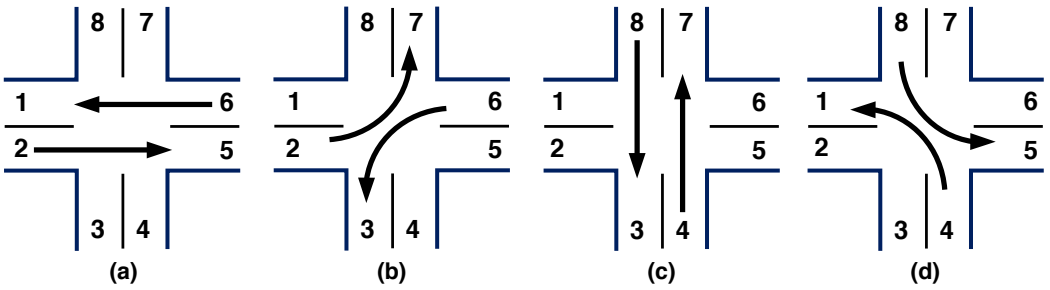


Figure 2.1: A typical intersection with eight movements and 4 admissible phases.

entry links and exit links together characterize the boundary of a system. This model can also take garages into account by modeling each garage as an entry link plus an exit link.

Given two links $i, j \in \mathcal{L}$ incident to the same intersection, link i is called a downstream link of j (or equivalently, i is an upstream link of j) if the end node of link i is the same as the start node of link j. We use $\mathcal{D}(i)$ and $\mathcal{U}(i)$ to denote the set of all the downstream and upstream links of each link i, respectively. Moreover, the link pair (i, j) forms a *movement* of vehicles. A collection of non-conflicting movements is called an *admissible phase* of an intersection. For each intersection, based on its scheduling policy, at each time slot exactly one of the admissible phases is chosen to have the right of way. Let $I_{i,j}(t)$ be the indicator function denoting whether $Q_{i,j}$ is scheduled at the corresponding intersection at time t. Figure 2.1 shows a standard intersection with eight movements and four admissible phases. In this typical intersection, each link has two upstream links and two downstream links. For ease of explanation, we assume that vehicles can only go straight or turn left, but cannot turn right, in this example. Each movement (i, j) has an associated queue $Q_{i,j}$ holding incoming vehicles. In other words, we assume that there exists a

separate queue for each left-turn and through movement. We assume that each queue has infinite capacity such that there is no overflow or blockage at each intersection.

2.3.2 Arrivals, Services, and Routing

External vehicles enter the system only via entry links. For any entry link i and its downstream link $j \in \mathcal{D}(i)$, let $\{A_{i,j}(t)\}_{t\geq 0}$ be an i.i.d. sequence of external arrivals at $Q_{i,j}$ with average external arrival rate $\lambda_{i,j} > 0$, and $A_{i,j}(t) \leq A_{\max}$ at any time t. For any non-entry link i and $j \in \mathcal{D}(i)$, we simply let $A_{i,j}(t) = 0$ for all t and hence $\lambda_{i,j} = 0$. For ease of later discussion, we also define $\lambda_i := \sum_{j\in\mathcal{D}(i)} \lambda_{i,j}$ to be the total external arrival rate through each link i. Similarly, let $\{S_{i,j}(t)\}_{t\geq 0}$ be an i.i.d. sequence of potential service rates of the movement (i, j), with average service rate $\mu_{i,j}$, for each movement $(i, j) \in \mathcal{M}$. We assume that $S_{i,j}(t) \leq S_{\max}$, for any movement (i, j) and any time t. $S_{i,j}(t)$ captures the variation in the passage time required by different vehicles. Since $S_{i,j}(t)$ depends on instantaneous conditions such as vehicle speed and driver behavior, it is difficult for the traffic scheduler to obtain information about potential service rates. Therefore, we presume that the traffic scheduler only has the information of *average service rate*, which is often called *saturation flow* in the transportation community. The average service rate of a movement is roughly proportional to the number of lanes of that movement [18].

In the multi-hop model, vehicles are routed in a probabilistic manner. When a vehicle enters a link i, it joins a downstream link $j \in \mathcal{D}(i)$ independently with probability $r_{i,j}$, with $\sum_{j\in\mathcal{D}(i)} r_{i,j} = 1$. We assume that $r_{i,j} > 0$, for all movements $(i, j) \in \mathcal{M}$. Let $R_{i,j}(t)$ denote the proportion of vehicles that join $Q_{i,j}$ from among the vehicles entering link i at time t, with $0 \leq R_{i,j}(t) \leq 1$. Since each vehicle chooses its route independently, $\mathbb{E}[R_{i,j}(t)] = r_{i,j}$ for any time t by the basic properties of multinomial random variables. Note that the above model of arrivals, service, and

routing is similar to that of the classic open Jackson network.

2.3.3 Scheduling and Switch-Over Delay

For each intersection, based on its scheduling policy, at each time slot exactly one of the admissible phases is chosen to have the right of way. Let $I_{i,j}(t)$ be the indicator function denoting whether $Q_{i,j}$ is scheduled at the corresponding intersection at time t. Therefore, for each intersection $v \in \mathcal{V}$, we can use a $|\mathcal{M}_v|$-dimensional binary vector to represent the scheduled phase of the intersection. Let $\mathcal{I}_v$ be the collection of the schedule vectors of all the admissible phases at the intersection v. Then, under a scheduling policy, each intersection v determines $\mathbf{I}_v(t) \in \mathcal{I}_v$ at each time t.

In order to guarantee absolute safety, non-zero time delay is inserted for an intersection to switch the right of way from the current admissible phase to the next. Such lost service time during traffic signal change is modeled as a *switch-over delay*, during which all the movements at the intersection are prohibited and hence the throughput is zero. For simplicity, we assume that the switch-over delay is T_S slot(s) for all the intersections. For each intersection, the time slots between any two consecutive switch-over events are further grouped into a *frame*. The frame sizes indicate how frequently an intersection switches between admissible phases.

In the proposed research, each intersection is either a *fixed-time intersection* or a *connected intersection*. A fixed-time intersection simply follows the weighted round-robin policy with the weights determined a priori according to long-term average traffic demands. In contrast, a connected intersection dynamically makes scheduling decisions based on real-time information obtained via connected-vehicle technology, such as queue length. We use $\mathcal{V}_F$ and $\mathcal{V}_C$ to denote the set of fixed-time intersections and connected intersections, respectively.

2.3.4 Queueing Dynamics

An intersection is said to be *active* if it is not in switch-over. Let $X_{i,j}(t)$ be the indicator function that the movement (i,j) is active in time slot t. Then, for any $t \geq 0$ and for any movement $(i,j) \in \mathcal{M}$ with link $i \in \mathcal{L}_{\text{entry}}$, the queue length is updated by

$$Q_{i,j}(t+1) = Q_{i,j}(t) - \big(S_{i,j}(t)I_{i,j}(t)X_{i,j}(t) \wedge Q_{i,j}(t)\big) + A_{i,j}(t), \tag{2.1}$$

where $(x \wedge y) := \min\{x, y\}$. Note that the second term on the right-hand side of (2.1) represents the number of vehicles that actually leave $Q_{i,j}$ in time slot t and the third term is the external arrivals at $Q_{i,j}$ in time slot t. On the other hand, for any movement $(i,j) \in \mathcal{M}$ with link $i \notin \mathcal{L}_{\text{entry}}$, we have

$$Q_{i,j}(t+1) = Q_{i,j}(t) - \big(S_{i,j}(t)I_{i,j}(t)X_{i,j}(t) \wedge Q_{i,j}(t)\big) \tag{2.2}$$

$$+ \sum_{m:(m,i)\in\mathcal{M}} \Big(S_{m,i}(t)I_{m,i}(t)X_{m,i}(t) \wedge Q_{m,i}(t)\Big)R_{i,j}(t). \tag{2.3}$$

Note that (2.3) represents the total number of vehicles coming from the upstream links of i in time slot t.

2.4 Problem Formulation

To study throughput-optimality, we first need to characterize the capacity region of a multi-hop transportation system.

Definition 1. *A multi-hop transportation system is strongly stable under a scheduling policy π if*

$$\limsup_{T\to\infty} \frac{1}{T} \sum_{\tau=0}^{T-1} \sum_{(i,j)\in\mathcal{M}} \mathbb{E}\big[Q_{i,j}(\tau)\big] < \infty. \tag{2.4}$$

When so, we say that the policy π stabilizes the system.

Next, we define the feasible external arrival rate vectors:

Definition 2. *Given a multi-hop transportation system $\mathcal{G}$, an external arrival rate vector $\boldsymbol{\lambda} = (\lambda_i)_{i\in\mathcal{L}}$ is* feasible *if there exists a scheduling policy under which the system is strongly stable with $\boldsymbol{\lambda}$.*

We can define the capacity region as follows:

Definition 3. *The capacity region is defined as the closure of the set of all feasible external arrival rate vectors $\boldsymbol{\lambda}$.*

To explicitly characterize the capacity region, we first obtain the *effective arrival rate*, which includes both external arrivals as well as arrivals from upstream links of each link, and then characterize the capacity region. Let λ_i^* be the effective arrival rate of link i. According to our model, we have $\lambda_i^* = \lambda_i$ for all $i \in \mathcal{L}_{\text{entry}}$. For any link $j \in \mathcal{L} \setminus \mathcal{L}_{\text{entry}}$, the effective arrival rate is determined by $\lambda_j^* = \sum_{i:j\in\mathcal{D}(i)} \lambda_i^* r_{i,j}$. Let $\boldsymbol{\lambda}^* = (\lambda_i^*)_{i\in\mathcal{L}}$ be the effective arrival rate vector, and $\mathbf{R} = (r_{i,j})_{i,j\in\mathcal{L}}$ the routing probability matrix. Then, we can write the system of traffic equations in matrix form:

$$\boldsymbol{\lambda}^* = \boldsymbol{\lambda} + \mathbf{R}^\intercal \boldsymbol{\lambda}^*, \tag{2.5}$$

where $\mathbf{R}^\intercal$ is the transpose of the routing probability matrix. Note that (2.5) is similar to the system of traffic equations of an open Jackson network. Let $\mathbf{1}$ be an $|\mathcal{L}| \times |\mathcal{L}|$ identity matrix. It is easy to verify that (2.5) has as a unique solution $\boldsymbol{\lambda}^* = (\mathbf{1} - \mathbf{R}^\intercal)^{-1}\boldsymbol{\lambda}$, where $(\mathbf{1} - \mathbf{R}^\intercal)$ is invertible (Section 2.1 in [19]).

For each fixed-time intersection v, let $\xi_v \in (0, 1)$ be the average fraction of time in which the intersection v is in switch-over. Let Λ be the set of all the external arrival rate vectors $\boldsymbol{\lambda}$ for which the following conditions hold: (i) For each fixed-time

intersection v, there exists $\epsilon > 0$ and a vector $\mathbf{\Sigma}_v = (\Sigma_{i,j})_{(i,j)\in\mathcal{M}_v}$ in the convex hull of $\mathcal{I}_v$ such that the effective arrival rates satisfy

$$\xi_v \mu_{i,j} \Sigma_{i,j} > \lambda_i^* r_{i,j} + \epsilon, \quad \forall (i,j) \in \mathcal{M}_v, \tag{2.6}$$

i.e. there is at least a small service margin for every movement at v. (ii) For each connected intersection $v \in \mathcal{V}_C$ there exists $\epsilon > 0$ and a vector $\mathbf{\Sigma_v} = (\Sigma_{i,j})_{(i,j)\in\mathcal{M}_v}$ in the convex hull of $\mathcal{I}_v$ such that

$$\mu_{i,j} \Sigma_{i,j} > \lambda_i^* r_{i,j} + \epsilon, \quad \forall (i,j) \in \mathcal{M}_v \,. \tag{2.7}$$

Let $\overline{\Lambda}$ denote the closure of Λ. The following provides a sufficient condition for feasibility of an arrival rate vector.

Theorem 1. *For a multi-hop transportation system with switch-over delay, an external arrival rate vector $\boldsymbol{\lambda} = (\lambda_i)_{i\in\mathcal{L}}$ is feasible if $\boldsymbol{\lambda} \in \Lambda$.*

Proof. This can be proved by finding an appropriate fixed-time policy for each connected intersection. By Theorem 1 in [5], we know that given any $\boldsymbol{\lambda} \in \Lambda$, there exists a fixed-time policy for each connected intersection such that the whole system is strongly stable. Hence, $\boldsymbol{\lambda}$ is feasible if $\boldsymbol{\lambda} \in \Lambda$. □

Next, we provide a necessary condition for feasibility.

Theorem 2. *For a multi-hop transportation system with switch-over delay, if $\boldsymbol{\lambda} \notin \overline{\Lambda}$, then there exists no policy under which the system is strongly stable.*

Proof. This is a direct result of Theorem 1 in [5]. □

The capacity region can be characterized as follows:

Theorem 3. *Given a multi-hop transportation system $\mathcal{G}$ with switch-over delay, the capacity region of $\mathcal{G}$ is $\overline{\Lambda}$.*

In this research, we focus on the interior of the capacity region and define throughput-optimality as follows:

Definition 4. *Given a multi-hop transportation system $\mathcal{G}$, a scheduling policy π is said to be* throughput-optimal *if the system is strongly stable under π for any external arrival rate vector $\boldsymbol{\lambda} \in \Lambda$.*

2.5 Scheduling for Throughput Optimality

In this section, we introduce our scheduling policy for connected intersections and show the main results regarding throughput-optimality under switch-over delay. To begin with, we define *pressure* as follows:

Definition 5. *For any time t, the pressure of a movement $(i,j) \in \mathcal{M}$ is defined as the difference between the queue length of (i,j) and the weighted average of the queue lengths of (j,k) for every $k \in \mathcal{D}(j)$, i.e.*

$$W_{i,j}(t) := Q_{i,j}(t) - \sum_{k:k\in\mathcal{D}(j)} r_{j,k} Q_{j,k}(t). \tag{2.8}$$

In addition, for any intersection v, the pressure of any admissible phase $\mathbf{I}_v = (I_{i,j}) \in \mathcal{I}_v$ is defined as $\sum_{i,j\in\mathcal{M}_v} \mu_{i,j} I_{i,j} W_{i,j}(t)$.

We also introduce a useful definition:

Definition 6. *A scheduling policy π is said to be* max-pressure-at-switch-over *if π always schedules the phase with the maximum pressure at each switch-over event.*

2.5.1 Proposed Scheduling Policy

To achieve throughput-optimality under switch-over delay, we propose the *Biased Max-Pressure (B-MP) scheduling policy* as shown in the following Algorithm 1.

In B-MP, frames are further grouped into consecutive *superframes*. At the beginning of a superframe, the duration of the superframe is calculated by (2.9). Whenever a connected intersection switches, it always switches to a phase with the maximum pressure, and therefore B-MP is max-pressure-at-switch-over. Under B-MP, a connected intersection will only switch under two conditions: (i) at the beginning of each superframe, or (ii) when conditions (2.10) and (2.11) specified below are satisfied. From conditions (2.10)-(2.11), B-MP only makes a switch when the maximum pressure is larger than the pressure of the current phase by a certain portion. Condition (2.10) can be interpreted as adding a bias factor favoring the pressure of the current phase, and hence the name B-MP. This bias for the current phase is to prevent the traffic signal from significant capacity loss due to frequent switch-overs.

Moreover, within a superframe, each connected intersection under B-MP can make scheduling decisions independently based on only the local queue length information. Therefore, B-MP is fully distributed within each superframe and the coordination among the connected intersections is minimal. We use t_k to denote the beginning of the k-th superframe, with $t_0 := 0$. Let $T_k := t_{k+1} - t_k$ be the length of the k-th superframe. Let M_k^v be the number of switch-over events in the k-th superframe, for each connected intersection v. Since each superframe may contain a different number of frames at different connected intersections, we use $t_{k,l}^v$ to denote the time of the l-th switch-over at intersection v in the k-th superframe, setting $t_{k,0}^v := t_k$.

Algorithm 1 Biased Max-Pressure Policy (B-MP)

1: Fix $\beta \in (0,1)$. At time $t = t_k$, set the length of the k-th superframe as:

$$T_k := \Big(\sum_{(i,j)\in\mathcal{M}} Q_{i,j}(t_k) \Big)^{\beta}, \tag{2.9}$$

and begin the next superframe at $t_{k+1} := t_k + T_k$.

2: Find the phase with the largest pressure at current time t,

$$\mathbf{I}_v^*(t) \in \arg\max_{\mathbf{I}\in\mathcal{I}_v} \sum_{(i,j)\in\mathcal{M}_v} \mu_{i,j} I_{i,j} W_{i,j}(t).$$

Ties are broken arbitrarily.

3: If $\mathbf{I}_v^*(t) \neq \mathbf{I}_v^*(t-1)$, initiate switch-over over the next T_S slots, and then apply the new schedule $\mathbf{I}_v^*(t)$ for one slot. Else, directly apply $\mathbf{I}_v^*(t)$ for one slot.

4: For any $t \in [t_{k,l}^v, t_{k,l+1}^v)$ in the rest of the k-th superframe, find the phase $\mathbf{I}_v^*(t)$ that has the largest pressure. If the intersection is not in switch-over at time t, the intersection makes a switch if the following condition is satisfied:

$$\Big(1 + B_v(t_{k,l}^v)\Big)\Big(\sum_{(i,j)\in\mathcal{M}_v} \mu_{i,j} I_{i,j}^*(t-1) W_{i,j}(t) \Big)^+ \tag{2.10}$$

$$< \Big(\sum_{(i,j)\in\mathcal{M}_v} \mu_{i,j} I_{i,j}^*(t) W_{i,j}(t) \Big)^+, \tag{2.11}$$

where x^+ is a shorthand for $\max\{x, 0\}$, and $B_v(\cdot)$ is the "bias function" defined as

$$B_v(t) := \zeta T_S \min\left\{ 1, \left(\left[\sum_{(i,j)\in\mathcal{M}_v} W_{i,j}(t) \right]^+ \right)^{-\alpha} \right\} \tag{2.12}$$

with $\alpha \in (0,1)$ and $\zeta > 0$. Else, stay at the current phase.

5: Repeat Step 3 and 4 until the end of the k-th superframe.

6: At $t = t_{k+1}$, go back to step 1 and repeat the above procedure for the next superframe.

2.5.2 Proof of Throughput-Optimality

To study system stability, we consider the queue length update over one superframe. Define $\Delta Q_{i,j}(t_k) := Q_{i,j}(t_{k+1}) - Q_{i,j}(t_k)$. By (2.1), for any movement (i,j) with link $i \in \mathcal{L}_{\text{entry}}$, we have

$$\Delta Q_{i,j}(t_k) \tag{2.13}$$

$$= -\sum_{t=t_k}^{t_{k+1}-1} \Big(S_{i,j}(t) I_{i,j}(t) X_{i,j}(t) \wedge Q_{i,j}(t) \Big) + \sum_{t=t_k}^{t_{k+1}-1} A_{i,j}(t), \tag{2.14}$$

where $(x \wedge y) := \min\{x, y\}$. Note that the first term of (2.14) represents the number of vehicles that actually leave $Q_{i,j}$ during the k-th superframe, and the second term is the total number of external arrivals at $Q_{i,j}$ in the k-th superframe. On the other hand, by (2.2) and (2.3), for any movement $(i,j) \in \mathcal{M}$ with link $i \notin \mathcal{L}_{\text{entry}}$, we have

$$\Delta Q_{i,j}(t_k) = -\sum_{t=t_k}^{t_{k+1}-1} \Big(S_{i,j}(t) I_{i,j}(t) X_{i,j}(t) \wedge Q_{i,j}(t) \Big) \tag{2.15}$$

$$+ \sum_{t=t_k}^{t_{k+1}-1} \sum_{m:(m,i)\in\mathcal{M}} \Big(S_{m,i}(t) I_{m,i}(t) X_{m,i}(t) \wedge Q_{m,i}(t) \Big) R_{i,j}(t). \tag{2.16}$$

Note that (2.16) represents the total number of vehicles coming from the upstream links of i during the k-th superframe.

To study the throughput performance, we analyze the Lyapunov drift over one superframe (see [20] for more details on multi-slot drift analysis). Define a Lyapunov function

$$L(\mathbf{Q}(t)) := \mathbf{Q}(t)^{\intercal}\mathbf{Q}(t) = \sum_{(i,j)\in\mathcal{M}} Q_{i,j}(t)^2, \tag{2.17}$$

where $\mathbf{Q}(t)^\intercal$ is the transpose of the queue length vector. Define the Lyapunov drift over the k-th superframe as $\Delta L(t_k) := L(\mathbf{Q}(t_{k+1})) - L(\mathbf{Q}(t_k))$. Then, we have

$$\Delta L(t_k) = 2\mathbf{Q}(t_k)^\intercal \Delta \mathbf{Q}(t_k) + \Delta \mathbf{Q}^\intercal \Delta \mathbf{Q}(t_k), \tag{2.18}$$

where $\Delta \mathbf{Q}(t_k) := \mathbf{Q}(t_{k+1}) - \mathbf{Q}(t_k)$. Given $\mathbf{Q}(t_k)$, the size of the k-th superframe is known and therefore the conditional drift over the k-th superframe is well-defined. Note that it is actually not straightforward to calculate the conditional drift over one superframe for the following reasons:

- For any intersection, there could be multiple frames, and hence multiple phases are scheduled in a stochastic sequence in one superframe.
- Different intersections could possibly have totally different frame sizes in the same superframe.
- Given the queue length information at the beginning of a superframe, it is still not clear when switch-over will be triggered and which phase will be scheduled at each intersection, since the arrival and service processes are stochastic.

Despite the above challenges, the conditional drift over one superframe can still be upper bounded for the max-pressure-at-switch-over policies.

Lemma 1. *Given any $\boldsymbol{\lambda} \in \Lambda$, under any max-pressure-at-switch-over policy with superframe structure, the conditional drift over one superframe is upper bounded*

as

$$\mathbb{E}\big[\Delta L(t_k) \,\big|\, \mathbf{Q}(t_k)\big] \leq -2\epsilon T_k \sum_{(i,j)\in\mathcal{M}} W_{i,j}(t_k)^+ \tag{2.19}$$

$$+ C_1 \sum_{v\in\mathcal{V}_C} \mathbb{E}\big[M_k^v \,\big|\, \mathbf{Q}(t_k)\big] \Bigg(\sum_{(i,j)\in\mathcal{M}_v} W_{i,j}(t_k)^+ \Bigg) \tag{2.20}$$

$$+ C_2 \sum_{v\in\mathcal{V}_F} \sum_{(i,j)\in\mathcal{M}_v} W_{i,j}(t_k)^+ + C_3 T_k^2 + C_4 T_k \tag{2.21}$$

where C_1, C_2, C_3 and C_4 are finite positive constants and $x^+ := \max\{x, 0\}$.

Proof. With the max-pressure-at-switch-over property, we are able to quantify the pressure of the scheduled phases at any $t \in [t_k, t_{k+1})$ even if the scheduling decision of each frame is not known. The complete proof is provided in the Appendix A.1. □

Remark 1. *Note that (2.19) represents the negative drift required for system stability. Also note that (2.20) and the first term of (2.21) represent the loss of service due to switch-over at connected intersections and fixed-time intersections, respectively. The second and third terms of (2.21) stand for the service loss due to possible emptiness of the scheduled queues.*

Remark 2. *Note that in (2.20) the service loss due to switch-over is basically a direct sum of the service loss contributed by each connected intersection. In other words, the performances of any two connected intersections are completely decoupled. Due to this feature, Lemma 1 still holds if different connected intersections follow different max-pressure-at-switch-over policies with superframe structure.*

To show that B-MP is throughput-optimal, we introduce a sufficient condition for strong stability in the following lemma.

Lemma 2. *For any max-pressure-at-switch-over scheduling policy with superframe determined by (2.9), if there exist some constants $B_0 > 0$, $\epsilon_0 > 0$ such that the conditional drift satisfies*

$$\mathbb{E}\big[\Delta L(t_k) \,\big|\, \mathbf{Q}(t_k)\big] \leq B_0 - \epsilon_0 \Big(\sum_{(i,j)\in\mathcal{M}} Q_{i,j}(t_k) \Big)^{1+\beta}, \tag{2.22}$$

then we have

$$\limsup_{T\to\infty} \frac{1}{T} \sum_{t=0}^{T-1} \sum_{(i,j)\in\mathcal{M}} \mathbb{E}\big[Q_{i,j}(t)\big] < \infty. \tag{2.23}$$

Proof. Define $H(t_k) := \sum_{t=0}^{T_k-1} \sum_{(i,j)\in\mathcal{M}} Q_{i,j}(t_k + t)$. Then, we have

$$\begin{aligned} H(t_k) \leq & \sum_{t=0}^{T_k-1} \sum_{i\in\mathcal{L}_{\text{entry}}, j\in\mathcal{D}(i)} \Big(Q_{i,j}(t_k) + \sum_{s=0}^{T_k-1} A_{i,j}(t_k + s) \Big) \\ & + \sum_{t=0}^{T_k-1} \sum_{i\in\mathcal{L}_{\text{int}}, j\in\mathcal{D}(i)} Q_{i,j}(t_k). \end{aligned}$$

After taking conditional expectation of $H(t_k)$, we have

$$\mathbb{E}\big[H(t_k) \,\big|\, \mathbf{Q}(t_k)\big] \tag{2.24}$$

$$\leq T_k^2 \Big(\sum_{i\in\mathcal{L}_{\text{entry}}, j\in\mathcal{D}(i)} \lambda_i^* r_{i,j} \Big) + T_k \Big(\sum_{(i,j)\in\mathcal{M}} Q_{i,j}(t_k) \Big) \tag{2.25}$$

$$\leq B_1 \Big(\sum_{(i,j)\in\ \mathcal{M}} Q_{i,j}(t_k) \Big)^{1+\beta} \tag{2.26}$$

where $B_1 = 1 + \sum_{i\in\mathcal{L}_{\text{entry}}} \lambda_i^* r_{i,j}$ is a positive constant independent of $\mathbf{Q}(t_k)$. Then,

by (2.22),

$$\mathbb{E}\big[\Delta L(t_k) \,\big|\, \mathbf{Q}(t_k)\big] \leq B_0 - \frac{\epsilon_0}{B_1}\,\mathbb{E}\big[H(t_k) \,\big|\, \mathbf{Q}(t_k)\big]. \tag{2.27}$$

By summing (2.27) over all the superframes, we have

$$\sum_{k\geq 0}\mathbb{E}\big[\Delta L(t_k) \,\big|\, \mathbf{Q}(t_k)\big] \leq \sum_{k\geq 0}\Big(B_0 - \frac{\epsilon_0}{B_1}\,\mathbb{E}\big[H(t_k) \,\big|\, \mathbf{Q}(t_k)\big]\Big). \tag{2.28}$$

Given a finite initial condition $\mathbf{Q}(0)$, we have $L(0) < \infty$ and $\sum_{k\geq 0}\mathbb{E}\big[\Delta L(t_k)\big|\mathbf{Q}(t_k)\big] \geq -L(0)$. Hence, we conclude that

$$\limsup_{T\to\infty}\frac{\sum_{t=0}^{T-1}\mathbb{E}\big[\sum_{(i,j)\in\mathcal{M}}Q_{i,j}(t)\big]}{T} \leq \frac{B_1\big(B_0+L(0)\big)}{\epsilon_0} < \infty.$$

□

Next, since Lemma 1 involves both queue length and pressure, we provide a useful inequality between total queue length and total pressure as follows.

Lemma 3. *For any queue length vector $\mathbf{Q} = (Q_{i,j})$ and its corresponding pressure vector $\mathbf{W} = (W_{i,j})$, there exists a constant $\delta > 0$ such that*

$$\sum_{(i,j)\in\mathcal{M}} {W_{i,j}}^{+} \geq \delta\Big(\sum_{(i,j)\in\mathcal{M}} Q_{i,j}\Big). \tag{2.29}$$

Proof. We provide a sketch of the proof. We first construct a new system by adding several dummy links and dummy movements to the original system and show that the new system is strongly connected, and that the corresponding routing matrix is invertible. By applying the Perron-Frobenius Theorem to the routing matrix, we obtain a strictly positive eigenvector with a positive eigenvalue. Based on the

eigenvector properties, we show that there must exist a constant $\delta > 0$ such that the inequality (2.29) holds. The complete proof is provided in Appendix A.2. □

Note that B-MP is a max-pressure-at-switch-over policy and therefore Lemma 1 holds under the B-MP policy. To characterize the number of switch-over events in one superframe under the B-MP policy, we provide an upper bound on the size of each frame as follows.

Lemma 4. *Under the B-MP policy, there exists a constant $C_5 > 0$ such that for every sample path, the length of each frame is lower bounded as*

$$T^v_{k,l} \geq C_5 B_v(t^v_{k,l}) \Big(\sum_{(i,j)\in\mathcal{M}_v} W_{i,j}(t^v_{k,l})^+ \Big). \tag{2.30}$$

Proof. The proof is provided in Appendix A.3. □

With Lemma 4, we are ready to provide a lower bound on the number of switch-over events in one superframe under B-MP.

Lemma 5. *For any $k \geq 0$, given $\mathbf{Q}(t_k)$, for any intersection v under the B-MP policy with bias function defined by (2.12), we have for every sample path,*

$$M^v_k \Big(\sum_{(i,j)\in\mathcal{M}_v} W_{i,j}(t_k)^+ \Big) = o\bigg(\Big(\sum_{(i,j)\in\mathcal{M}} Q_{i,j}(t_k) \Big)^{1+\beta} \bigg). \tag{2.31}$$

Proof. The proof is provided in Appendix A.4. □

We are ready to show that B-MP is throughput-optimal.

Theorem 4. *The B-MP policy is throughput-optimal for any* $\alpha \in (0,1)$, $\beta \in (0,1)$.

Proof. Since B-MP is a max-pressure-at-switch-over policy with superframe structure, Lemma 1 holds under B-MP. Therefore, by Lemma 3 and the fact that $W_{i,j}(t)^+ \leq Q_{i,j}(t)$ for any movement (i,j) and any time t, we have

$$\mathbb{E}\big[\Delta L(t_k) \,\big|\, \mathbf{Q}(t_k)\big] \leq -2\epsilon\delta_0 T_k \sum_{(i,j)\in\mathcal{M}} Q_{i,j}(t_k) \tag{2.32}$$

$$+ C_1 \sum_{v\in\mathcal{V}_C} \mathbb{E}\big[M_k^v \,\big|\, \mathbf{Q}(t_k)\big]\Big(\sum_{(i,j)\in\mathcal{M}_v} W_{i,j}(t_k)^+\Big) \tag{2.33}$$

$$+ C_2 \sum_{v\in\mathcal{V}_F} \sum_{(i,j)\in\mathcal{M}_v} Q_{i,j}(t_k) + C_3 T_k^2 + C_4 T_k. \tag{2.34}$$

By Lemma 5 and the choice of T_k, we know $T_k \sum_{(i,j)\in\mathcal{M}} Q_{i,j}(t_k)$ is the dominating term in (2.32)-(2.34). Therefore, there exists a constant $B > 0$ such that

$$\mathbb{E}\big[\Delta L(t_k) \,\big|\, \mathbf{Q}(t_k)\big] \leq B - \epsilon\delta_0\Big(\sum_{(i,j)\in\mathcal{M}} Q_{i,j}(t_k)\Big)^{1+\beta}. \tag{2.35}$$

By Lemma 2, we know that the system is strongly stable under the B-MP policy for any external arrival rate $\boldsymbol{\lambda} \in \Lambda$. Hence, the B-MP policy is throughput-optimal.

□

Remark 3. *By Theorem 4, B-MP can achieve throughput-optimality for any choice of* α *between 0 and 1. The choice of* α *can indeed affect the average delay performance, and is a topic that is not addressed in this research.*

Remark 4. *The parameter* β *determines the superframe size for coordination among the intersections. To minimize the coordination overhead, it is recommended that* β

be close to 1.

2.6 Extensions of Biased Max-Pressure Policy

2.6.1 Weighted Queue Length

The concept of pressure can be further generalized by using *"weighted queue lengths"*:

Definition 7. *Let $q_{i,j} > 0$ be the predetermined weight factor of movement (i,j). For each movement (i,j), we define the "weighted queue length" as $\hat{Q}_{i,j}(t) := q_{i,j}Q_{i,j}(t)$, for all t. Then, the generalized pressure is defined as*

$$\hat{W}_{i,j}(t) := \hat{Q}_{i,j}(t) - \sum_{k:k\in\mathcal{D}(j)} r_{j,k}\hat{Q}_{j,k}(t). \tag{2.36}$$

If one substitutes $\hat{W}_{i,j}(t)$ for $W_{i,j}(t)$, the B-MP policy continue to remain throughput-optimal:

Theorem 5. *The B-MP policy using the generalized pressure in Definition 7 is still throughput-optimal for any $\alpha \in (0,1)$, any $\beta \in (0,1)$.*

Proof. This can be proved by considering the drift of a Lyapunov function: $\hat{L}(\mathbf{Q}(t)) = \sum_{(i,j)\in\mathcal{M}} q_{i,j}Q_{i,j}(t)^2$. The rest of the proof is similar to that of Theorem 4 and hence omitted due to space limitation. □

One important usage of weighted queue lengths is to design a capacity-aware version of the B-MP policy that mitigates the *queue overflow effect* due to finite queue capacity. Queue overflow often occurs when the system operates under oversaturated traffic (even if only for a short period of time). The overflow effect can lead to significant service loss as well as severe delay. Given the information about queue

capacity, we can choose $q_{i,j}$ appropriately for each movement (i,j) to reduce the chance of queue overflow. For example, choosing $q_{i,j}$ inversely proportional to the queue capacity of $Q_{i,j}$ is suggested in [21]. In Section 4.6, we provide an example of applying weighted queue length in simulation.

2.6.2 Estimated Queue Length With Bounded Error

In networked transportation systems, it might be difficult or expensive to obtain precisely accurate queue length information, due to latency in communication, or random errors in sensor detection. Let $Q^{\dagger}_{i,j}(t)$ and $W^{\dagger}_{i,j}(t)$ be the estimated queue length and the corresponding pressure, respectively. If the estimation error of queue length is always upper bounded, then the B-MP is still throughput-optimal with the estimated queue length. We still consider the Lyapunov function $L(\mathbf{Q}(t)) = \sum_{(i,j)\in\mathcal{M}} Q_{i,j}(t)^2$ and the corresponding drift conditioned on $\mathbf{Q}^{\dagger}_{i,j}(t_k)$. Then, we have the following upper bound on the conditional drift:

Lemma 6. *Given any $\boldsymbol{\lambda} \in \Lambda$, under the B-MP policy using estimated queue length $(Q^{\dagger}_{i,j}(t))$, if there exists a constant $B > 0$ such that $\left|Q_{i,j}(t) - Q^{\dagger}_{i,j}(t)\right| \leq B$ for all (i,j) and all t, the conditional drift over one superframe is upper bounded as:*

$$\mathbb{E}\big[\Delta L(t_k) \,\big|\, \mathbf{Q}^{\dagger}(t_k)\big] \leq -2\epsilon T_k \sum_{(i,j)\in\mathcal{M}} W^{\dagger}_{i,j}(t_k)^{+} \tag{2.37}$$

$$+ C^{\dagger}_1 \sum_{v\in\mathcal{V}_C} \mathbb{E}\big[M^v_k \,\big|\, \mathbf{Q}^{\dagger}(t_k)\big] \left(\sum_{(i,j)\in\mathcal{M}_v} W^{\dagger}_{i,j}(t_k)^{+} \right) \tag{2.38}$$

$$+ C^{\dagger}_2 \sum_{v\in\mathcal{V}_F} \sum_{(i,j)\in\mathcal{M}_v} W^{\dagger}_{i,j}(t_k)^{+} + C^{\dagger}_3 T_k^2 + C^{\dagger}_4 T_k \tag{2.39}$$

where $C^{\dagger}_1$, $C^{\dagger}_2$, $C^{\dagger}_3$ and $C^{\dagger}_4$ are finite positive constants.

Proof. The proof is similar to that of Lemma 1. The main differences are: (i) Since the drift is now conditioned on $\mathbf{Q}^\dagger(t_k)$ instead of $\mathbf{Q}(t_k)$, the estimation error introduces an extra term in $\mathbb{E}\Big[\mathbf{Q}(t_k)^\intercal \Delta\mathbf{Q}(t_k) \mid \mathbf{Q}^\dagger(t_k)\Big]$. Due to the boundedness of estimation error, this extra term is at most of the same order as T_k. (ii) For connected intersections, B-MP using $\mathbf{Q}^\dagger(t_k)$ makes scheduling decisions based on $\mathbf{W}^\dagger(t_k)$. Therefore, B-MP is max-pressure-at-switch-over in terms of $\mathbf{W}^\dagger(t_k)$ instead of $\mathbf{W}(t_k)$. Since $Q_{i,j}(t) - Q^\dagger_{i,j}(t) \in [-B, B]$, we also have $W_{i,j}(t) - W^\dagger_{i,j}(t) \in [-2B, 2B]$, for all (i,j) and all t. As a result, the bounded error in pressure only affects the coefficients of the existing terms in the original drift expression. The complete proof is provided in Appendix A.5. □

Now, we are ready to prove that B-MP is throughput-optimal with estimated queue lengths.

Theorem 6. *If there exists a constant $B > 0$ such that $\Big|Q_{i,j}(t) - Q^\dagger_{i,j}(t)\Big| \leq B$ for all (i,j) and all t, then B-MP is still throughput-optimal using estimated queue lengths $(Q^\dagger_{i,j}(t))$.*

Proof. First, we have $Q_{i,j}(t) - Q^\dagger_{i,j}(t) \in [-B, B]$ and $W_{i,j}(t) - W^\dagger_{i,j}(t) \in [-2B, 2B]$, for all (i,j) and all t, Also, Lemma 3 holds regardless of the scheduling policy. Therefore, we can rewrite the upper bound in Lemma 6 as

$$\mathbb{E}\big[\Delta L(t_k) \,\big|\, \mathbf{Q}^\dagger(t_k)\big] \leq -2\epsilon\delta_0 T_k \sum_{(i,j)\in\mathcal{M}} Q^\dagger_{i,j}(t_k) \tag{2.40}$$

$$+ C_1^\ddagger \sum_{v\in\mathcal{V}_C} \mathbb{E}\big[M_k^v \,\big|\, \mathbf{Q}^\dagger(t_k)\big] \Big(\sum_{(i,j)\in\mathcal{M}_v} W^\dagger_{i,j}(t_k)^+ \Big) \tag{2.41}$$

$$+ C_2^\ddagger \sum_{v\in\mathcal{V}_F} \sum_{(i,j)\in\mathcal{M}_v} W^\dagger_{i,j}(t_k) + C_3^\ddagger T_k^2 + C_4^\ddagger T_k, \tag{2.42}$$

where $C_1^\ddagger$, $C_2^\ddagger$, $C_3^\ddagger$, $C_4^\ddagger$ are finite positive constants. Furthermore, with a slight modification of the proof we know that Lemma 4 and Lemma 5 still hold when $\mathbf{W}(t_k)$ is replaced by $\mathbf{W}^\dagger(t_k)$ under B-MP. By the same argument as that in the proof of Theorem 4, we know that $-2\epsilon T_k \sum_{(i,j)\in\mathcal{M}} Q_{i,j}^\dagger(t_k)$ is the dominating term in (2.40)-(2.42). Therefore, there must exist a constant $B^\dagger > 0$ such that

$$\mathbb{E}\big[\Delta L(t_k) \,\big|\, \mathbf{Q}^\dagger(t_k)\big] \leq B^\dagger - \epsilon\delta_0 \Bigg(\sum_{(i,j)\in\mathcal{M}} Q_{i,j}^\dagger(t_k) \Bigg)^{1+\beta}. \tag{2.43}$$

By a similar procedure as in Lemma 2, we know that (2.43) is also a sufficient condition for strong stability. Hence, we conclude that B-MP remains throughput-optimal when the error in queue lengths is bounded. □

From Theorem 4, we know that B-MP is also robust to estimation error in queue length information.

2.6.3 Limitations on Green Period

Conventionally, the timing plan of traffic signals includes a minimum green time to accommodate the vehicle startup delay. Under the B-MP policy, the minimum green time can be easily incorporated by introducing a minimum frame size $T_{G,\min} > T_S$. Then, (2.30) in Lemma 4 would become

$$T_{k,l}^v \geq \max\Bigg\{ T_{G,\min}, C_5 B_v(t_{k,l}^v) \Big(\sum_{(i,j)\in\mathcal{M}_v} W_{i,j}(t_{k,l}^v)^+ \Big) \Bigg\}. \tag{2.44}$$

With a slight modification of the proof of Lemma 5, the B-MP policy with a minimum frame size still remains throughput-optimal. On the other hand, a maximum green time is sometimes applied in the actuated version of fixed-time policy to avoid excessive delays of minor roads. While this can also be included in B-MP by in-

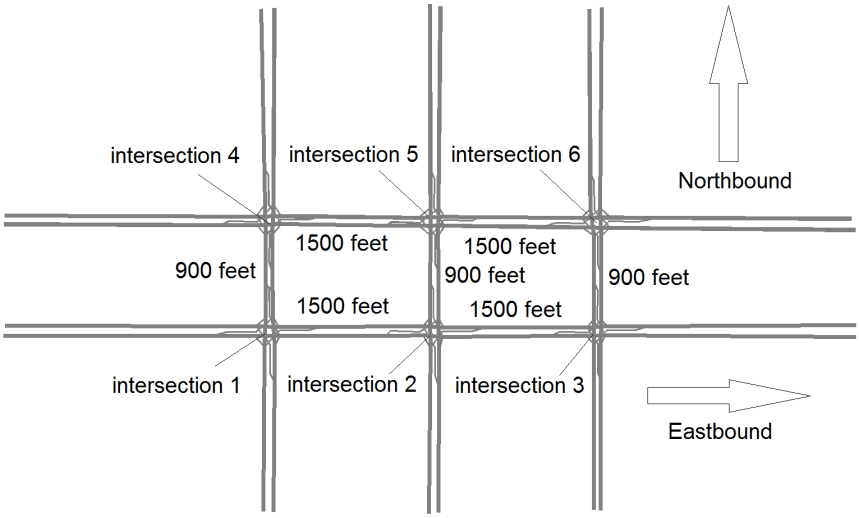


Figure 2.2: System topology in VISSIM.

troducing a maximum frame size $T_{G,\max}$, setting a maximum frame size can result in loss of system throughput since the fraction of time spent on switch-over would always be greater than or equal to $\frac{T_S}{T_{G,\max}}$.

2.6.4 Simulation Results

2.7 Simulations

We evaluate the proposed policy in VISSIM [22], which is a standard microscopic traffic simulator for transportation systems. In addition to the built-in features for conventional traffic signal control, VISSIM also provides programming integration with MATLAB to support user-customizable traffic control algorithms.

We consider a system of six signalized intersections as shown in Figure 2.2. In total, there are 10 entry links (4 major entries from the East and the West along with 6 minor entries from the North and the South) and 10 exit links. The number of lanes of each through-traffic link and left-turn link are 3 and 1, respectively.

Conforming to the official statistics [18], the saturation flow of each link is set to be 1900 vehicles per hour per lane. Vehicles enter the system from the entry links and are routed towards an exit link in a probabilistic manner. We set the routing probabilities to be 0.2 and 0.8 for left-turn movement and through movement, respectively. We use λ_{E}, λ_{W}, λ_{N}, and λ_{S} to denote the arrival rates of the entry links coming from the East, West, North, and South, respectively. We use the default driver behavior and lane-change model provided in VISSIM. The speed limit of each vehicle is 40 miles per hour. Each intersection has four admissible phases as described in Figure 2.1. Throughout the simulation, we choose the slot time to be 1 second which is sufficient for updating the scheduling decisions. The switch-over delay is set to be 5 seconds, which includes an amber period of 3 seconds and an all-red period of 2 seconds. An important feature of our VISSIM simulation is that we consider the effect of finite buffer size. When a link is fully occupied by vehicles, VISSIM will prohibit the entry of new vehicles, either from the external or from upstream links, and hence lower the throughput.

We compare the B-MP policy against the conventional fixed-time policy, Max-Pressure (MP) policy, and the Variable Frame-Based Max-Weight (VFMW) policy. For the fixed-time policy, the timing plan is calculated by Synchro [23], which is a widely-used optimization tool for timing plan design in transportation research. Throughout the simulation, we assume that the fixed-time policy has perfect knowledge of the average traffic statistics of each link, and is therefore able to optimize the timing plan accordingly. For VFMW, we choose the frame size to be $T_S + \left(\sum_{(i,j)\in\mathcal{M}_v} Q_{i,j}(t_k)\right)^{0.9}$ as suggested in [24]. For the B-MP policy, we choose $\alpha = 0.01$ and $\beta = 0.99$ as discussed in Section 2.5.2. To mitigate possible queue overflow due to finite queue capacity, we use weighted queue lengths with $q_{i,j} = 3$ for through-traffic queues and $q_{i,j} = 1$ for left-turn queues, as discussed in Section 2.6.1.

First, we consider the following arrival traffic pattern:

Scenario 1: $\lambda_{\mathrm{E}} = \lambda_{\mathrm{W}} = \bar{\lambda}$ and $\lambda_{\mathrm{N}} = \lambda_{\mathrm{S}} = 0.5 \cdot \bar{\lambda}$ (veh/hr).

Under this traffic pattern, the maximum achievable $\bar{\lambda}$ is about 2600 veh/hr according to the traffic equations given by (2.5). The total simulation time is 1800 seconds. Figure 2.3 shows the total number of vehicles in the system with $\bar{\lambda} = 2400$ under the four policies. We observe that B-MP indeed achieves the smallest total queue length while the total queue length is much larger under the other three policies.

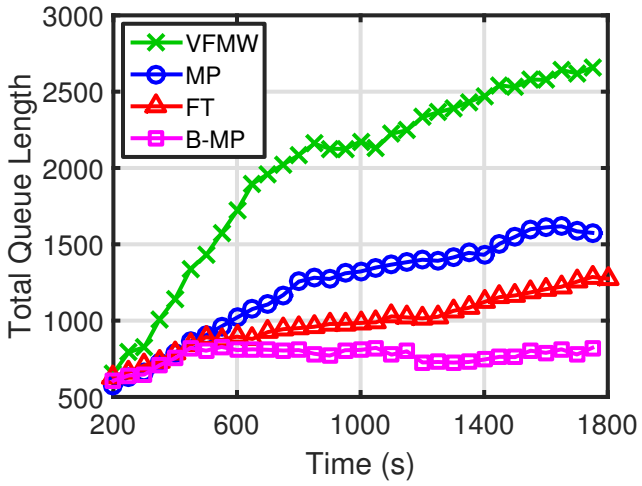


Figure 2.3: Total queue length of the system under the four policies with $\bar{\lambda} = 2400$.

Next, we measure the performance with $\bar{\lambda}$ between 1200 and 2800. Figures 2.4(a) and 2.4(b) show the system throughput and average delay for different arrival rates. Note that the average delay here is defined as the difference between the actual travel time and the travel time without any stoppages at the intersections. In Figure 2.4(a), we see that under B-MP the throughput grows linearly with the arrival rate for $\bar{\lambda}$ up

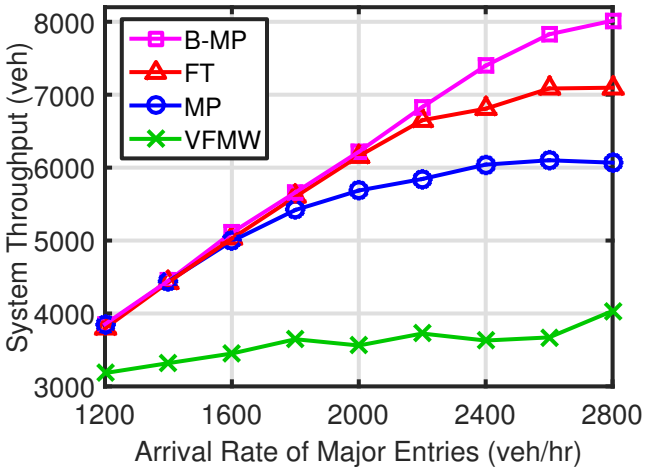
to 2600. At $\bar{\lambda} = 2800$, the throughput under B-MP gets saturated simply because the arrival rate is already beyond the capacity region. Concerning the fixed-time policy, it can support $\bar{\lambda}$ only up to 2200 due to the capacity loss resulting from the switch-over delay. Both MP and VFMW suffer from severe capacity loss due to frequent switching of traffic signals. In Figure 2.4(b), the B-MP still achieves the smallest delay for every $\bar{\lambda}$. For the heavy traffic condition with $\bar{\lambda} = 2600$, compared to the fixed-time policy with perfect knowledge of traffic statistics, B-MP reduces the average delay by more than 40% without any arrival rate information. For VFMW, we only show the average delay for $\bar{\lambda}$ below 1800 because it performs much more poorly than the other three policies for $\bar{\lambda}$ above 2000.

Next, we consider time-varying arrival rates.

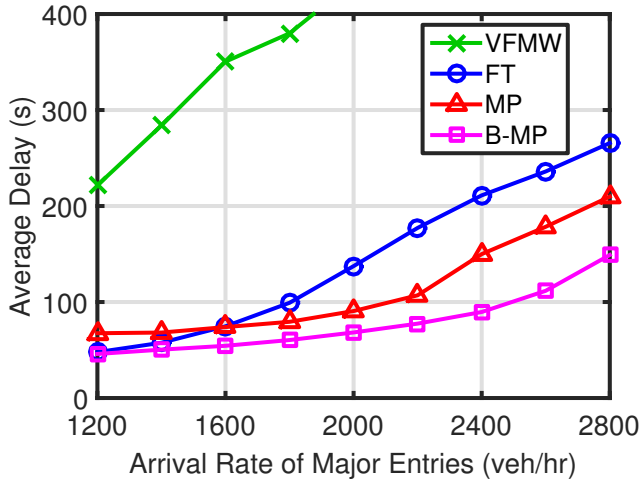
Scenario 2:

- 0 s to 1200 s: $(\lambda_{\mathrm{W}}, \lambda_{\mathrm{E}}, \lambda_{\mathrm{N}}, \lambda_{\mathrm{S}}) = (2000, 2000, 1000, 1000)$.
- 1201 s to 2400 s: $(\lambda_{\mathrm{W}}, \lambda_{\mathrm{E}}, \lambda_{\mathrm{N}}, \lambda_{\mathrm{S}}) = (2500, 1500, 1500, 500)$.
- 2401 s to 3600 s: $(\lambda_{\mathrm{W}}, \lambda_{\mathrm{E}}, \lambda_{\mathrm{N}}, \lambda_{\mathrm{S}}) = (1500, 2500, 500, 1500)$.

Note that the total arrival rate of the whole system remains the same under the above traffic pattern. Figure 2.5 shows the total queue length under the three policies. Here we omit the VFMW policy simply because it has a much larger total queue length. Again, B-MP still achieves the smallest total queue length at any time. It is notable that the total queue length under B-MP does not change much under the time-varying pattern. In contrast, the fixed-time policy suffers from much more congestion during the period 1200 s to 3600 s. This is because the fixed-time policy optimizes its timing plan based on the average arrival rates and thus fails to accommodate traffic dynamics. Similar to Figure 2.3, MP still performs quite poorly due to the service loss incurred by the switch-over delay.



(a) System throughput



(b) Average delay

Figure 2.4: Delay and throughput performance under the four policies for different arrival rates.

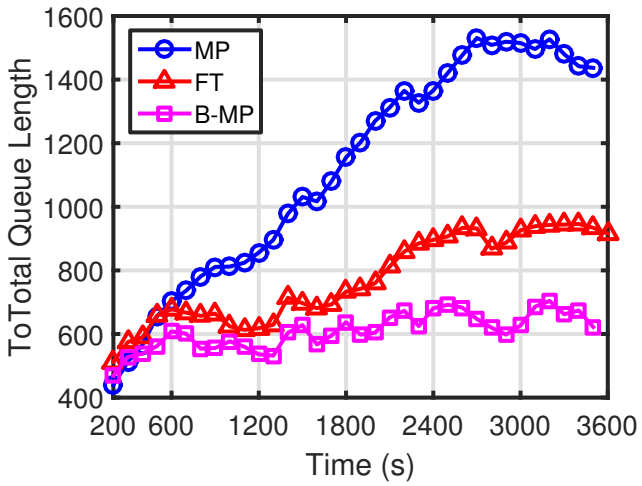
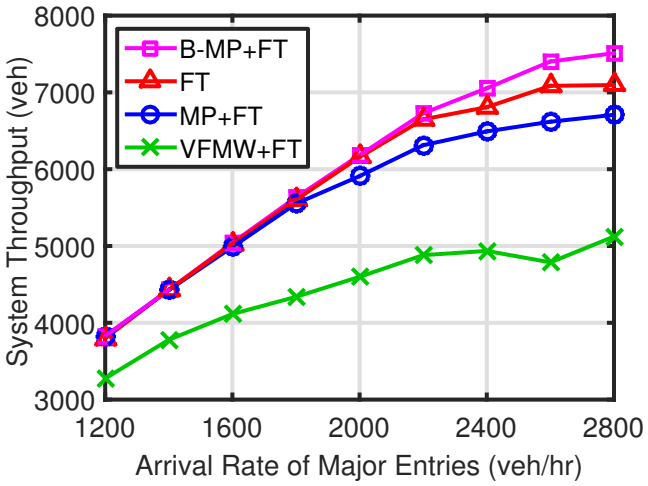
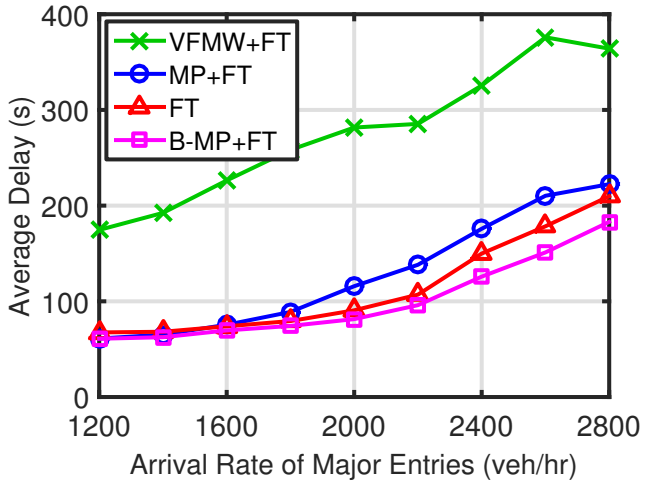


Figure 2.5: Total queue length under time-varying traffic.

Last, we consider a partially-connected system where three of the intersections are connected under a user-customized policy (B-MP, MP, or VFMW) and the rest are fixed-time intersections as usual. Figures 2.6(a) and 2.6(b) show the average delay and system throughput of the partially-connected system for different arrival rates. Compared to the pure fixed-time system, even partial inclusion of the B-MP policy still provides improvement in both throughput and average delay. Also, B-MP still outperforms the other two policies by a large margin in the partially-connected system. Through the above simulation, we see that B-MP indeed provides significant improvement over the other three popular policies.



(a) System throughput



(b) Average delay

Figure 2.6: Delay and throughput performance under the four policies in the partially-connected system.

2.8 Summary

In this chapter, we study the scheduling problem for networked transportation systems with switch-over delay. We propose a distributed scheduling policy that

is throughput-optimal with switch-over delay without requiring knowledge of traffic demands. Moreover, the proposed policy still remains optimal when there are both fixed-time intersections and connected intersections in the overall system. Hence, the proposed policy can still perform well in partially-connected systems. Simulation results show that the proposed policy indeed outperforms the other current policies.

3. QOE-OPTIMAL SCHEDULING FOR WIRELESS VIDEO DELIVERY[1]

3.1 Overview

In recent years, video streaming applications have been demanding more and more resource in wireless networks. According to the latest report from Cisco [26], video-centric services are projected to increase 13-fold and occupy nearly three-fourths of global mobile data traffic in the near future. However, upgrade of wireless network capacity can hardly catch up with the explosive mobile data traffic. Therefore, better scheduling algorithms are required for service providers to serve more customers without sacrificing user satisfaction.

From the perspective of service providers, scheduling policies are conventionally designed to meet the performance requirement of a general wireless network, such as average throughput, latency, delay jitter, etc. However, these statistics fail to directly characterize real user experience in enjoying video service. Hence, various research works have been carried out to study quality of experience (QoE) in video streaming applications. Much effort has been dedicated to quantifying subjective user experience and constructing analytical models based on different experiment setups, such as [27, 28, 29]. In general, QoE can be affected by several factors, such as playback smoothness, mean video quality, temporal variation in quality, etc. Among these elements, playback interruption has been shown to be the dominant factor of QoE performance [28, 29]. Therefore, we define QoE by the duration of video interruption during playback process for wireless on-demand video streams.

Playback interruption of a single video stream has been studied extensively in

[1]Part of this chapter is reprinted with permission from "Heavy-Traffic Analysis of QoE Optimality for On-Demand Video Streams Over Fading Channels" by Ping-Chun Hsieh and I-Hong Hou in Proc. of IEEE INFOCOM 2016 [25].

recent literature. In [30], the probability of interruption-free video playback is analyzed for variable bit-rate video over wireless channels with variable data rate. By modelling a playback buffer as a M/D/1 queue, [31] provides a bound on interruption probability for media streams over Markovian channels. Likewise, Xu *et al.* [32] and Anttonen *et al.* [33] provide explicit results of the distribution of video interruption based on different queueing models. Similarly, [34] presents an online algorithm to adaptively control playback buffer underflow and overflow based on large deviation theory. Moreover, by applying diffusion approximation to a G/G/1 queue, Luan *et al.* [35] characterize the dynamics of a video playback buffer under a threshold-based buffer management scheme. The common focus of these works is to provide an indicator to make the best tradeoff between initial prefetching delay and playback buffer emptiness. However, these results only work for a single video stream and thus can hardly be applied to a wireless network.

Regarding scheduling for QoE of multiple video streams, [36] and [37] provide a flow-level analytical framework to study the effect of flow dynamics on playback interruption and average throughput. [38] proposes an online algorithm based on Proportional-Fair scheduling to achieve fairness among video users while maintaining required throughput. In [39], a modified version of Proportional-Fair scheduling has been presented to reduce video inter-frame delay for wireless LTE networks. To offer better average rate guarantees, Bhatia *et al.* [40] design a scheduling policy which exploits slow-fading variation of wireless channels. In a multi-cast wireless network, [41] proves by dynamic programming that a Max-Weight like policy is throughput optimal. To improve video-rate-based QoE, Li *et al.* [42] design a scheduling policy based on the head-of-line packet delay and packet deadlines. In [43], a resource allocation algorithm is proposed to maximize video-rate-based utility while maintaining fairness in term of buffer level. In [44], Li *et al.* propose a joint rate control and

scheduling algorithm to optimize rate-based network utility for scalable videos in a multi-cast wireless network. In [45], Anand and de Veciana propose a scheduling policy that achieves asymptotically optimal delay-based QoE. In [46], Joseph and de Veciana consider a more comprehensive QoE metric and propose the NOVA algorithm to asymptotically optimize QoE for a wireless network. Based on a similar decomposition approach as [46], Xiao *et al.* [47] propose an online algorithm to optimize QoE specifically for OFDMA wireless networks. One common feature of the above policies is that they aim to optimize QoE in the sense of long-term average performance, such as average video quality and average playback interruption. Moreover, [48] considers the short-term QoE performance by studying the diffusion limit. However, it only considers constant-bit-rate video streams in wireless networks where the channel qualities are static.

In this research, we address QoE optimality and propose online scheduling policies for wireless on-demand video streams. We are particularly interested in QoE performance under heavy-traffic conditions. Different from the prior efforts of [36, 37, 38, 39, 40, 41, 42, 43, 44, 45, 46, 47], this research addresses not only long-term average performance but also *short-term QoE performance* by studying diffusion limits. Different from the study of diffusion limit for a single video stream without scheduling in [35], we study diffusion limits for a network of multiple video streams and propose online scheduling policies. In [35], diffusion approximation for a G/G/1 queue is directly applicable since the arrival and departure processes of a playback buffer are given and completely independent of the buffer management scheme. By contrast, for a network of video streams, the arrival process of each video buffer is controlled by the scheduling policy, and hence it is not immediately clear how to apply diffusion limits. Despite this challenge, we are able to characterize the capacity region for QoE and show that the proposed policies achieve the whole capacity region based on diffu-

sion limits. Instead of assuming static channel qualities and constant-bit-rate videos as in [48], we study the dynamic behavior of playback process of variable-bit-rate videos over time-varying channels. This research can be summarized as follows:

- For ease of presentation, we start from a special case of constant-bit-rate videos over ON-OFF channels. We first consider long-term average playback interruption by studying the stability region and providing a polynomial-time algorithm for checking the stabilizability when channels are independent (Section 3.3).

- We study short-term QoE performance of playback interruption by applying diffusion limits. Specifically, we start by deriving a lower bound for total playback interruption for all scheduling policies and providing necessary conditions for the capacity region for QoE (Section 3.4.3). We then propose an online scheduling policy and explicitly characterize the distribution of playback interruption at any given time under the proposed policy (Section 3.4.4). We thereby show that the proposed policy achieves every interior point in the capacity region for QoE.

- The policy and the heavy-traffic analysis are then generalized for general i.i.d. fading channels and variable-bit-rate videos. The proposed policies are proved to remain optimal under some mild assumptions.

- We formulate a network utility maximization problem based on the QoE of each client. We show that our policy can achieve the optimal network utility by selecting proper parameters.

- We compare the proposed policies against three popular policies and show by simulation that our policy surpasses other three policies by a large margin in short-term QoE, despite that the long-term average duration of video interruption approaches zero asymptotically for all these policies. Moreover, we also implement

the proposed policy on a software-defined wireless testbed and demonstrate the performance via real video streaming applications.

Through analysis and extensive evaluation of the proposed policies, we thereby demonstrate the essential difference between QoE and the conventional QoS metrics for wireless networks.

The rest of the chapter is organized as follows. Section 4.2 describes the network model for analyzing QoE of on-demand video streams. Section 3.3 discusses the stability region and an algorithm for checking stabilizability for ON-OFF channels. In Section 3.4, we present an online scheduling policy for ON-OFF channels plus constant-bit-rate videos and prove that it is optimal in heavy traffic. We then extend the policy for fading channels and variable-bit-rate videos in Section 3.5. In Section 3.6, we formulate a network utility maximization problem based on QoE. Simulation and experimental results of the proposed policies as well as the counterparts are shown in Section 4.6 and 3.8, respectively. Finally, Section 4.7 concludes this chapter.

3.2 System Model

We consider a time-slotted wireless network consisting of a wireless access point (AP) and a group of N mobile clients denoted by $S_{tot} = \{1, 2, ..., N\}$. Each client is downloading an on-demand video which has been pre-stored by video service providers. The video content is partitioned into packets and streamed to clients via the AP and the wireless links. On the AP's side, we assume that the AP always has packets at hand for transmission to each video client. In other words, the throughput for the AP to acquire video content from video providers is assumed to be much larger than that between the AP and the mobile clients. We also assume that there is no network coding mechanism involved in the system. Thus, in each time slot, the AP can transmit data to at most one client.

In a wireless network, the quality of a wireless channel usually changes with time. To capture the variation of wireless channels, we model the wireless link of each client n as a discrete-time random process $r_n(t)$, which is i.i.d. across time and takes only non-negative integer values in a finite *data rate space* denoted by $\mathcal{R}$. If the AP schedules a transmission for client n at time t, it can deliver exactly $r_n(t)$ bits. For example, the IEEE 802.11a standard has a maximum physical data rate of 54 Mbit/s, and the data rate can also be adaptively reduced by applying different modulation and coding, depending on channel conditions. In this model, we make no assumption about the relationship between different wireless links. Therefore, unless stated otherwise, the channels of different clients are *not* required to be independent.

On the mobile clients' side, the received packets are first decoded and queued in a playback buffer, whose size is assumed to be infinite. For each client n, let $A_n(t)$ denote the total amount of received video content in bits until time t, and $B_n(t)$ be the amount of video content in bits stored in the playback buffer at time t. Next, we consider the playback process of *variable-bit-rate* videos. Specifically, each client n plays one video frame every k_n slots, and different frames of the same video may contain different number of bits. Suppose for client n, the j-th frame contains $F_n(j)$ bits for every $j \geq 1$. We assume that $F_n(j)$ are i.i.d. across j with mean q_n^* and variance $\sigma_{q,n}^2$. We define $q_n := q_n^*/k_n$ to be the average video playback rate per slot, which reflects the desired video resolution. Moreover, we assume that $F_n(j)$ is upper bounded by $q_{n,\max}$ for every frame j. Let $C_n(J)$ be the sum of the frame size up to the J-th frame, i.e. $C_n(J) := \sum_{j=1}^{J} F_n(j)$. Let $S_n(t)$ be the total number of frames that the client n plays up to t. Therefore, the total bits played up to time t is $C_n(S_n(t))$. At time slot t, the number of bits in the playback buffer of client n is

$$B_n(t) = A_n(t) - C_n(S_n(t)), \tag{3.1}$$

where we assume that $B_n(0) = 0$ for every client.

When the client is about to play a new video frame from the playback buffer, playback interruption might occur if there is no enough video content in the playback buffer. To check this condition, it is equivalent to check if the buffer $B_n(t)$ becomes negative after the video frame about to play is taken out of the buffer. Let $D_n(t)$ be the total number of slots in which video is interrupted by time t. Given $D_n(t-1)$, $\lfloor \frac{t-D_n(t-1)}{k_n} \rfloor$ is the total number of video frames that client n would play up to time t if video interruption does not happen at time t. Hence, $C_n(\lfloor \frac{t-D_n(t-1)}{k_n} \rfloor)$ represents the total number of bits played up to t if video interruption does not happen at time t. Then, $D_n(t)$ can be updated recursively by

$$D_n(t) = \begin{cases} D_n(t-1)+1, & \text{if } A_n(t) - C_n(\lfloor \frac{t-D_n(t-1)}{k_n} \rfloor) < 0 \\ D_n(t-1), & \text{otherwise} \end{cases} \tag{3.2}$$

From (3.2), we know that in each slot, $D_n(t)$ either stays unchanged or increases by 1. Define $B_n^*(t) := q_{n,\max} \lfloor \frac{B_n(t)}{q_{n,\max}} \rfloor$ to be the quantized version of $B_n(t)$ and let $e_n(t) := B_n^*(t) - B_n(t)$ be the quantization error of $B_n(t)$ with respect to $q_{n,\max}$. Note that $e_n(t)$ is bounded, i.e. $|e_n(t)| < q_{n,\max}$, for all $t \geq 0$. After the above manipulation, we know that $B_n^*(t) = 0$ if $D_n(t+1) - D_n(t) = 1$. Therefore, we have

$$B_n^*(t) = A_n(t) - C_n(S_n(t)) + e_n(t) \tag{3.3}$$

$$= (A_n(t) - q_n t) + q_n D_n(t) \tag{3.4}$$

$$- [C_n(S_n(t)) - q_n(t - D_n(t)) - e_n(t)]. \tag{3.5}$$

Define

$$Y_n(t) := C_n(S_n(t)) - q_n(t - D_n(t)) - e_n(t). \tag{3.6}$$

Since $t - D_n(t)$ is the total playback time with no interruption, then on average client n should already play $q_n(t - D_n(t))$ bits by time t. Since $C_n(S_n(t))$ is the total number of bits played up to time t, $Y_n(t)$ therefore reflects whether the amount of played video content matches the average playback rate of the variable-bit-rate videos. We also define that

$$X_n(t) := A_n(t) - q_n t. \tag{3.7}$$

Since on average client n plays q_n bits per time slot, then $X_n(t)$ reflects whether the amount of received data matches the average video playback rate. Now, we can summarize the basic properties as follows.

$$B_n^*(t) = X_n(t) - Y_n(t) + q_n D_n(t) \geq 0 \tag{3.8}$$

$$[D_n(t+1) - D_n(t)] \in \{0, 1\}, \quad D_n(0) = 0, \tag{3.9}$$

$$B_n^*(t)[D_n(t+1) - D_n(t)] = 0 \tag{3.10}$$

Based on (3.8)-(3.10), we can further connect $D_n(t)$ with $X_n(t)$ as follows.

Theorem 7. *Given $X_n(t)$ and $Y_n(t)$, there exists a unique pair of $B_n^*(t)$ and $D_n(t)$ that satisfies (3.8)-(3.10). Moreover, we have*

$$D_n(t) = \sup_{0 \leq \tau \leq t} (\max\{0, -\frac{X_n(\tau) - Y_n(\tau)}{q_n}\}). \tag{3.11}$$

Proof. This is a direct result of Theorem 6.1 in [19]. □ □

Remark 5. We know that $X_n(t)$ reflects whether the total amount of received data $A_n(t)$ matches the total number of played bits $q_n t$. Moreover, $Y_n(t)$ captures the fluctuation in video frame size. $X_n(t)$ and $Y_n(t)$ are sufficient to determine the buffer status and video interruption. Therefore, it is not surprising that there exists a unique pair of $B_n^*(t)$ and $D_n(t)$ that satisfies (3.8)-(3.11) as stated in Theorem 7.

Remark 6. To intuitively understand (3.11), consider an example where a client n receives a_n bits from the AP in every time slot and the video has a constant bit rate. Then, $X_n(t) = (a_n - q_n)t$, and $|Y_n(t)| \leq q_n k_n + |e_n(t)|$ is bounded.

- If $a_n > q_n$, then $X_n(t)$ grows linearly with t and by (3.11) we know $D_n(t)$ remains 0 for all t. In other words, there is no video interruption at all if the amount of received data per slot is always greater than the playback rate.

- If $a_n < q_n$, then $X_n(t)$ is always non-positive and decreases linearly with t. By (3.11) we know $D_n(t)$ grows almost linearly with t with some minor fluctuation due to the bounded term $Y_n(t)$. This corresponds to the fact that the video gets continually interrupted when the amount of received data per slot is always less than the required playback rate.

- If $a_n = q_n$, then both $X_n(t)$ and $Y_n(t)$ are 0 for all t. Thus, $D_n(t) = 0$ for all t.

In this research, QoE of each video stream is measured by its total duration of playback interruption. One usual way to assess QoE of a wireless network is through long-term average performance which is formally defined as follows.

Definition 8. *A video streaming system is said to be stabilizable if there exists a scheduling policy η such that* $\limsup_{t\to\infty} \frac{D_n(t)}{t} = 0$ *for all n, almost surely. Moreover, η is a stabilizing scheduling policy for QoE.* □

In other words, a wireless video-streaming system is stabilizable if the total duration of video interruption grows sublinearly after some finite time. We first consider the long-term average behavior of $Y_n(t)$. Since the frame sizes are i.i.d. with mean q_n^*, then by the Strong Law of Large Numbers for i.i.d. random variables, it can be easily shown that $\lim_{t\to\infty} \frac{Y_n(t)}{t} = 0$, almost surely, regardless of the scheduling policy. By (3.8), this implies that the fluctuation in video frame size does not affect the long-term average video interruption. Then, it can be easily shown that $\limsup_{t\to\infty} \frac{D_n(t)}{t} = 0$ if and only if the long-term average throughput of each client is at least q_n [48]. If $\liminf_{t\to\infty} \frac{A_n(t)}{t} < q_n$, then $X_n(t)$ will go to negative infinity as $t \to \infty$, almost surely. By (3.8), since $B_n^*(t)$ is always nonnegative, $\liminf_{t\to\infty} \frac{A_n(t)}{t} < q_n$ implies that $D_n(t)$ goes to infinity as $t \to \infty$, almost surely. Therefore, studying whether a system is stabilizable is equivalent to studying the capacity region of achievable throughput. However, this definition fails to characterize the behavior of the system in the heavy-traffic regime.

To fully characterize the growth of playback interruption with time, we study the dynamic behavior by using *diffusion limits* in the following parts of the chapter. Moreover, in Section 4.6, we will compare the proposed policy with other popular scheduling policies that are all stabilizing for QoE but are rather different in short-term performance.

To study the behavior of video interruption in the heavy-traffic regime, we consider the diffusion limit of $D_n(t)$, which is defined as

$$\hat{D}_n(t) := \lim_{k\to\infty} \frac{D_n(kt)}{\sqrt{k}}, \quad 0 \le t \le 1. \tag{3.12}$$

Similarly, we define

Table 3.1: Main notations used in the chapter.

Notation	Description
q_n^*	average frame size of client n
k_n	interval between two consecutive frames (if no interruption)
q_n	q_n^*/k_n, i.e. video playback rate of client n
$A_n(t)$	total number of bits received by client n up to time t
$S_n(t)$	total number of frames that client n plays up to t
$C_n(J)$	sum of the frame size of client n up to frame J
$X_n(t)$	$A_n(t) - q_n t$ (reflects if the arrival rate matches q_n)
$Y_n(t)$	$C_n(S_n(t)) - q_n(t - D_n(t)) - e_n(t)$, where $e_n(t)$ denotes a small bounded error
$D_n(t)$	total amount of video interruption seen by client n up to t
$\hat{X}_n(t), \hat{Y}_n(t)$	diffusion limits of $X_n(t)$ and $Y_n(t)$
$\hat{D}_n(t)$	diffusion limits of $D_n(t)$
$X(t)$	sum of $X_n(t)$ of all client n
$\hat{X}(t)$	diffusion limit of $X(t)$
$\hat{D}(t)$	$\sup_{0\le\tau\le t}(\max\{0, -\hat{X}(\tau)\})$
$r_n(t)$	data rate of client n at time t
$R(t,S)$	$\max\{r_n(t) : n \in S\}$
$R(t)$	highest data rate among all the clients at time t
$Z_n(t)$	$C_n(\lfloor \frac{t}{k_n} \rfloor) - q_n t$
$Z(t)$	sum of $Z_n(t)$ of all client n
$\hat{Z}_n(t), \hat{Z}(t)$	diffusion limits of $Z_n(t)$ and $Z(t)$

$$\hat{X}_n(t) := \lim_{k\to\infty} \frac{X_n(kt)}{\sqrt{k}}, \tag{3.13}$$

$$\hat{Y}_n(t) := \lim_{k\to\infty} \frac{Y_n(kt)}{\sqrt{k}}, \tag{3.14}$$

$$\hat{B}_n^*(t) := \lim_{k\to\infty} \frac{B_n^*(kt)}{\sqrt{k}}. \tag{3.15}$$

Given the properties in (3.7)–(3.10), we then have the following useful results.

Theorem 8. *Given $\hat{X}_n(t)$ and $\hat{Y}_n(t)$, there exists a unique pair of $(\hat{B}_n^*(t), \hat{D}_n(t))$ that satisfies*

$$\hat{B}_n^*(t) = \hat{X}_n(t) - \hat{Y}_n(t) + q_n \hat{D}_n(t) \geq 0 \tag{3.16}$$

$$\frac{d\hat{D}_n(t)}{dt} \geq 0, \quad \hat{D}_n(0) = 0 \tag{3.17}$$

$$\hat{B}_n^*(t) \frac{d\hat{D}_n(t)}{dt} = 0. \tag{3.18}$$

Moreover, $\hat{D}_n(t)$ can be expressed as

$$\hat{D}_n(t) = \sup_{0 \leq \tau \leq t} (\max\{0, -\frac{\hat{X}_n(t) - \hat{Y}_n(t)}{q_n}\}) \tag{3.19}$$

Proof. This is a direct result of Theorem 1 in [48]. □ □

Remark 7. (3.19) can be viewed as the diffusion limit version of (3.11).

Theorem 8 suggests a general recipe on how to study video interruption in the diffusion limit:

- Characterize $\hat{X}_n(t)$ based on the channel model and the scheduling policy.
- Characterize $\hat{Y}_n(t)$ based on the dynamics of video bit rate.
- Combine $\hat{X}_n(t)$ and $\hat{Y}_n(t)$ to derive $\hat{D}_n(t)$ based on (3.19).

For ease of presentation, we start from a special case of constant-bit-rate videos over ON-OFF channels in Section 3.3 and 3.4, and then extend the analysis for fading channels and variable-bit-rate videos in Section 3.5.

In order to distinguish the analysis on $\limsup_{t\to\infty} \frac{D_n(t)}{t}$ and that on $\hat{D}_n(t)$, we use *stability region* to denote the set of stabilizable systems, and *capacity region* to

denote the set of achievable vectors of $\hat{D}_n(t)$. A more formal definition of capacity region is introduced in Section 3.4.

For convenience, we summarize the main notations used in this chapter in Table 3.1.

3.3 Stability Region for ON-OFF Channels

We first consider a special case of ON-OFF channels, where transmission rate of each client can only be either zero or a positive value r^*, and therefore $\mathcal{R} = \{0, r^*\}$.

Recall that the AP can transmit data to at most one client in each time slot. In this case, the stability region for ON-OFF channels has been shown to be associated with a set of necessary and sufficient conditions [49, 50]. We summarize the results as follows.

Lemma 7. *[49, Theorem 1] Let W_n be the event that client n has an ON channel, i.e., $r_n(t) = r^*$. A video streaming system with ON-OFF channels is stabilizable if and only if the video playback rates $\{q_n\}$ satisfy the following equations:*

$$Pr\left[\bigcup_{n\in S} W_n\right] \geq \frac{1}{r^*}\sum_{n\in S} q_n, \quad \forall S \subseteq S_{tot}. \tag{3.20}$$

Remark 8. For a given subset S, (3.20) indicates that the total demand of the subset S should not exceed the maximum total channel resource of the subset S.

The above condition requires checking (3.20) for all subsets, which can be intractable. However, for the special case where the channel conditions of different clients are independent, we can derive a polynomial-time algorithm to check whether a system is stabilizable. The algorithm is described in the following theorem.

Theorem 9. *Let p_n be the probability that client n has an ON channel, and $p_n > 0$, $\forall n$. Suppose that the clients are sorted based on $\frac{q_n}{p_n}$ in descending order, i.e. $\frac{q_1}{p_1} \geq \frac{q_2}{p_2} \geq \cdots \geq \frac{q_N}{p_N}$. Denote by S_k the subset $\{1, ..., k\}$ of all clients. Then, the system is stabilizable if and only if*

$$1 - \prod_{n \in S_k} (1 - p_n) \geq \frac{1}{r^*} \sum_{n \in S_k} q_n, \quad 1 \leq k \leq N. \tag{3.21}$$

Moreover, the complexity of checking this condition is $O(N \log N)$. □

Proof. The proof can be found in Appendix B.1. □ □

3.4 Heavy-Traffic Analysis for ON-OFF Channels and Constant-Bit-Rate Videos

We are particularly interested in the situation where the set of video playback rates $\{q_n\}$ is on the boundary of the stability region, that is, under the *heavy-traffic* condition.

3.4.1 Heavy-Traffic Conditions for ON-OFF Channels

Recall that W_n denotes the event that client n has ON channel. In this section, we assume that,

$$Pr\left[\bigcup_{n=1}^{N} W_n\right] = \frac{1}{r^*} \sum_{n=1}^{N} q_n, \tag{3.22}$$

while for any subset $S \subset \{1, ..., N\}$,

$$Pr\left[\bigcup_{n \in S} W_n\right] > \frac{1}{r^*} \sum_{n \in S} q_n. \tag{3.23}$$

The constraint (3.23) corresponds to the *complete resource pooling* condition in [51]. The complete resource pooling condition guarantees that there is enough overlap in the channel resources of different clients. This technical condition enables us to characterize the system using one-dimensional Brownian motion as in [51].

3.4.2 Constant-Bit-Rate Videos

For constant-bit-rate videos, all the frames of the same video have exactly the same size q_n^*, for each client n. From Remark 6, we know that $|Y_n(t)| \leq q_n k_n + |e_n(t)|$ is bounded, regardless of the scheduling policy. Therefore, by the definition of diffusion limit, we have $\hat{Y}_n(t) = 0$, for all t, under any scheduling policy. By Theorem 8, we have

$$\hat{D}_n(t) = \sup_{0 \leq \tau \leq t} (\max\{0, -\frac{\hat{X}_n(t)}{q_n}\}). \tag{3.24}$$

This implies that for constant-bit-rate video streams, $\hat{X}_n(t)$ fully characterizes the behavior of video playback interruption in the diffusion limit. Therefore, we can focus on characterizing $\hat{X}_n(t)$ in the rest of this section.

3.4.3 A Lower-Bound of Capacity Region for QoE

We derive fundamental properties of $D_n(t)$ with ON-OFF channels under the heavy-traffic conditions. Let us define a random process

$$X(t) := \sum_{n=1}^{N} X_n(t) = \sum_{n=1}^{N} (A_n(t) - q_n t). \tag{3.25}$$

Let $\Delta X(t+1) := X(t+1) - X(t)$ be the amount of change in $X(t)$, for all $t \geq 0$. Regardless of the scheduling policy, the AP can deliver exactly r^* bits to some client n if at least one client in S_{tot} has an ON channel. Let $\gamma := Pr\left[\bigcup_{n=1}^{N} W_n\right]$ be the probability of the event that at least one client has an ON channel. Then, in each time slot t, $\sum_{n=1}^{N} A_n(t)$ increases by r^* with probability γ and stays the same with

probability $1-\gamma$, regardless of the scheduling policy. Therefore, we have

$$\Delta X(t+1) = \begin{cases} -\sum_{n=1}^{N} q_n, & \text{with probability } 1-\gamma \\ \\ r^* - \sum_{n=1}^{N} q_n, & \text{with probability } \gamma \end{cases} \tag{3.26}$$

Since the channels are i.i.d. across time, the equations in (3.26) hold regardless of time and thus $\Delta X(t)$ is i.i.d. across all time slots. Due to the heavy-traffic assumption given by (3.22), we further have

$$E[\Delta X(t)] = \gamma(r^* - \sum_{n=1}^{N} q_n) + (1-\gamma)(-\sum_{n=1}^{N} q_n) = 0$$

$$\text{Var}[\Delta X(t)] = \gamma(r^* - r^*\gamma)^2 + (1-\gamma)(r^*\gamma)^2 = \gamma(1-\gamma)(r^*)^2.$$

By the *functional central limit theorem* for i.i.d. random variables [19], we have the following important properties regarding the diffusion limit of $X(t)$.

Theorem 10. *Let* $\hat{X}(t) := \lim_{k\to\infty} \frac{X(kt)}{\sqrt{k}}$. *Then* $\hat{X}(t)$ *is a driftless Brownian motion with variance* σ_x^2, *where* $\sigma_x = r^*\sqrt{(\gamma(1-\gamma))}$. *Moreover, given* $\hat{X}(\tau)$, *for any* $\tau, t \geq 0$ *with* $\tau + t \leq 1$, $\hat{X}(\tau+t) - \hat{X}(\tau)$ *is a Gaussian random variable with zero mean and variance* $\sigma_x^2 t$. $\square$

Remark 9. By Theorem 10, we are able to fully characterize the distribution of $\hat{X}(t)$ while the original process $X(t)$ can be difficult to analyze. This manifests the benefit of taking the diffusion limit of the original process.

Similar to (3.24), we define

$$\hat{D}(t) := \sup_{0\leq\tau\leq t} (\max\{0, -\hat{X}(\tau)\}) \tag{3.27}$$

Since $\hat{X}(t)$ is a Brownian motion, we can thereby derive the distribution and important statistics of $\hat{D}(t)$ based on the following lemma.

Lemma 8. *[52, Section 1.6] Let $\Phi(x)$ be the cumulative distribution function (CDF) of a standard Gaussian random variable with zero mean and unit variance. The CDF of $\hat{D}(t)$ is given by*

$$Pr[\hat{D}(t) \leq x] = \Phi(\frac{x}{\sqrt{\sigma_x^2 t}}) - \Phi(\frac{-x}{\sqrt{\sigma_x^2 t}})$$

for all $x \geq 0$, $t \geq 0$. The expected value of $\hat{D}(t)$ is given by

$$E[\hat{D}(t)] = \int_0^\infty x \sqrt{\frac{2}{\pi \sigma_x^2 t}} \exp(-\frac{x^2}{2\sigma_x^2 t}) dx = \sqrt{\frac{2t\sigma_x^2}{\pi}}. \quad \square$$

Given the characteristics of $\hat{D}(t)$, we obtain a lower bound for dynamics of video interruption seen by the clients. We first introduce the concept of stochastic ordering as follows [19].

Definition 9. *Let $\hat{D}^{\eta_1}(t)$ and $\hat{D}^{\eta_2}(t)$ be two real-valued random processes under policies η_1 and η_2, respectively. We say that $\hat{D}^{\eta_1}(t) \leq_{st} \hat{D}^{\eta_2}(t)$ if*

$$Pr[\hat{D}^{\eta_1}(t) \geq x] \leq Pr[\hat{D}^{\eta_2}(t) \geq x], \tag{3.28}$$

for all $x \in \mathbb{R}$ and for any $t \in [0, 1]$. $\square$

Then, we further build the relationship between $\hat{D}(t)$ and $\hat{D}_n(t)$ by using of $\hat{X}(t)$

and $\hat{X}_n(t)$:

$$\hat{D}(t) = \sup_{0\leq\tau\leq t}(\max\{0, -\hat{X}(t)\}) \tag{3.29}$$

$$= \sup_{0\leq\tau\leq t}(\max\{0, -\sum_{n=1}^{N}\hat{X}_n(\tau)\}) \tag{3.30}$$

$$\leq_{st} \sum_{n=1}^{N}\sup_{0\leq\tau\leq t}(\max\{0, -\hat{X}_n(t)\}) = \sum_{n=1}^{N} q_n\hat{D}_n(t). \tag{3.31}$$

Motivated by (3.29)-(3.31), we define the *capacity region* for QoE in terms of the diffusion limits $\hat{D}(t)$ and $\hat{D}_n(t)$ as follows.

Definition 10. *A N-tuple vector $\lambda = [\lambda_1, \lambda_2, ..., \lambda_N]$ is said to be feasible if there exists a scheduling policy such that*

$$\hat{D}_n(t) \leq_{st} \frac{\lambda_n}{q_n}\hat{D}(t), \quad n = 1, 2, ..., N. \tag{3.32}$$

Then, the capacity region for QoE, denoted by Λ, is defined as the set of all feasible vectors λ. □

Remark 10. Note that the capacity region for QoE is defined in terms of the diffusion limits of video interruption time, instead of the arrival rate region considered in many studies on long-term average throughput (such as [53]). Regarding the capacity analysis of long-term average throughput in classical queueing theory, it has been widely known that the capacity region can be characterized by using stationary randomized policies. By contrast, in the capacity analysis for QoE studied in this research, it is not immediately clear how to characterize the capacity region based on diffusion limits or how to achieve the whole capacity region. This also manifests the difference between our study on QoE and the conventional studies on long-term

average throughput.

Theorem 11. *A feasible vector* $\lambda = [\lambda_1, \lambda_2, ..., \lambda_N]$ *with* $\lambda_n \geq 0$*, for all* n*, must satisfy* $\sum_{n=1}^{N} \lambda_n \geq 1$. □

Proof. By Definition 10, if λ is feasible, then $q_n \hat{D}_n(t) \leq_{st} \lambda_n \hat{D}(t)$, for every client n. Therefore, $\sum_{n=1}^{N} q_n \hat{D}_n(t) \leq_{st} \sum_{n=1}^{N} \lambda_n \hat{D}(t)$. By (3.29)-(3.31), we have $\sum_{n=1}^{N} \lambda_n \geq 1$. □ □

3.4.4 Scheduling Policy

Now, we introduce a scheduling policy for constant-bit-rate videos over ON-OFF channels and show that it achieves every point in the interior of the capacity region for QoE.

Joint Channel-Deficit Policy (JCD):

In each time slot, the AP schedules the client n with the smallest value of $w_n X_n(t)$ among those clients with $r_n(t) = r^*$, where w_n is a predetermined weight factor. □

To prove that JCD policy achieves every point in the interior of the capacity region for QoE, we first establish the *state space collapse* property to characterize the diffusion limit $\hat{X}_n(t)$ of each individual client.

Theorem 12. *Let* w_n *be the weight for client* n *which is predetermined by the AP. For any pair of clients* n, m *in* S_{tot}*, we have* $w_n \hat{X}_n(t) = w_m \hat{X}_m(t)$*. Moreover, we can obtain that*

$$\hat{X}_n(t) = \frac{\frac{1}{w_n}}{\sum_{m=1}^{N} \frac{1}{w_m}} \hat{X}(t) = \beta_n \hat{X}(t), \tag{3.33}$$

where $\beta_n := \frac{\frac{1}{w_n}}{\sum_{m=1}^{N}\frac{1}{w_m}}$ *and* $\sum_{n=1}^{N}\beta_n = 1$. □

Proof. To show state-space collapse, we start from a fluid system induced by $X_n(t)$. Next, we consider a Lyapunov function and show the random process of the fluid system is positive recurrent. The complete proof can be found in Appendix B.2. □ □

Based on Theorem 12, we show that the JCD policy achieves every point in the capacity region by choosing proper $\{w_n\}$.

Theorem 13. *Given any vector* $\lambda = [\lambda_1, \lambda_2, ..., \lambda_N]$ *which satisfies* $\lambda_n > 0$, $\forall n$, *and* $\sum_{n=1}^{N}\lambda_n \geq 1$, *JCD policy achieves* $\hat{D}_n(t) = \frac{\beta_n}{q_n}\hat{D}(t) \leq_{st} \frac{\lambda_n}{q_n}\hat{D}(t)$ *with* $\beta_n := \frac{\frac{1}{w_n}}{\sum_{m=1}^{N}\frac{1}{w_m}}$ *and* $\sum_{n=1}^{N}\beta_n = 1$, *and thus the vector* λ *is feasible. Moreover,*

$$E[\hat{D}_n(t)] = \sqrt{\frac{2t\sigma_x^2}{\pi}}\frac{\beta_n}{q_n} = \sqrt{\frac{2t\sigma_x^2}{\pi}}\frac{\frac{1}{q_n w_n}}{\sum_{m=1}^{N}\frac{1}{w_m}}. \tag{3.34}$$

Proof. From (3.24) and (3.33), we know that

$$\hat{D}_n(t) = \sup_{0\leq\tau\leq t}(\max\{0, -\frac{\hat{X}_n(t)}{q_n}\}) \tag{3.35}$$

$$= \sup_{0\leq\tau\leq t}(\max\{0, -\frac{\beta_n\hat{X}(t)}{q_n}\}) = \frac{\beta_n}{q_n}\hat{D}(t). \tag{3.36}$$

By assigning $w_n = \frac{1}{\lambda_n}$ for all n, we have $\beta_n = \frac{\frac{1}{w_n}}{\sum_{m=1}^{N}\frac{1}{w_m}} = \frac{\lambda_n}{\sum_{m=1}^{N}\lambda_m} \leq \lambda_n$, where the last inequality holds since $\sum_{m=1}^{N}\lambda_m \geq 1$. Therefore, we conclude that $\hat{D}_n(t) = \frac{\beta_n}{q_n}\hat{D}(t) \leq_{st} \frac{\lambda_n}{q_n}\hat{D}(t)$. Moreover, (3.34) follows directly from (3.36) and Lemma 8. □ □

Remark 11. From an engineering point of view, by choosing w_n for each client, we can control the total playback interruption seen by each client and hence differentiate levels of service among all clients. In real applications, $\{w_n\}$ can be determined by a proper pricing scheme given by service providers. One simple example is paid VIP membership: choose a proper pair $\mu_1, \mu_2 > 0$ with $\mu_1 > \mu_2$, assign $w_n = \mu_1$ if client n is a VIP member; otherwise, assign $w_n = \mu_2$. This scheme enables differentiated service in QoE.

Here, we do not consider the boundary points which have $\lambda_n = 0$ for some n. In practice, we can get as close as possible to the boundary points by technically assigning extremely large w_n to our policy. Theorem 13 also characterizes a sufficient condition for the capacity region.

Theorem 14. *Given a vector* $[\lambda_1, \lambda_2, \dots]$ *with* $\lambda_n > 0, \forall n$*, the vector is feasible if and only if* $\sum_{n=1}^{N} \lambda_n \geq 1$. $\square$

3.5 Heavy-Traffic Analysis For General Fading Channels and Variable-Bit-Rate Videos

In this section, we further relax the assumptions that channels are ON-OFF and videos have constant bit rates in every frame, and study the playback process with general i.i.d. fading channels and variable-bit-rate videos.

3.5.1 General Fading Channels and Heavy-Traffic Conditions

We consider general i.i.d. fading channels where the data rate space $\mathcal{R}$ can consist of any number of different rates, as described in Section 4.2. Recall that the AP can transmit data to at most one client in each time slot. Unlike the case of ON-OFF channels, the stability region cannot be determined by a simple set of conditions

as those in Lemma 7. Instead, we impose the following conditions to simplify the analysis.

Recall that q_n^* and q_n are the mean frame size and the mean video consumption rate in bits per slot, respectively. Let $R(t,S) := \max\{r_n(t) : n \in S\}$ and $R(t)$ be the shorthand for $R(t, S_{tot})$. In other words, $R(t)$ represents the highest data rate at time t among all the clients. We assume that

$$E\left[R(t)\right] = \sum_{n=1}^{N} q_n, \tag{3.37}$$

and

$$E\left[R(t) \cdot \mathbb{I}\left\{R(t,S) = R(t)\right\}\right] > \sum_{n \in S} q_n, \tag{3.38}$$

for all $S \subsetneq S_{tot}$, where $\mathbb{I}\{\cdot\}$ is an indicator function. (3.37) serves as the heavy-traffic condition for general fading channels, i.e. the condition where the maximum system-wide channel resource exactly matches the total video playback rate. Besides, similar to (3.23), (3.38) corresponds to the complete resource pooling condition for general fading channels. We also note that these conditions reduce to (3.22) and (3.23) when $\mathcal{R} = \{0, r^*\}$, i.e. ON-OFF channels. It is easy to check these two conditions are sufficient for a system to be stabilizable. Further, it characterizes a portion of the boundary of the stability region, as it is not possible to increase q_n for any client n without making the system unstabilizable.

Similar to Section 3.4, we define $X(t) := \sum_{n=1}^{N} X_n(t) = \sum_{n=1}^{N}(A_n(t) - q_n t)$ and $\Delta X(t) := X(t) - X(t-1)$. Under the heavy-traffic condition given by (3.37), $\forall t > 0$

we have

$$E[\Delta X(t)] = \sum_{n=1}^{N} E[A_n(t) - A_n(t-1)] - \sum_{n=1}^{N} q_n \tag{3.39}$$

$$\leq E[R(t)] - \sum_{n=1}^{N} q_n \leq 0, \tag{3.40}$$

regardless of the scheduling policy. To obtain a lower bound of capacity region as in Section 3.4, we first consider a special class of policies, denoted by Π^*, which only schedule clients with the largest $r_n(t)$ at any time $t > 0$. This class of policies still need to determine which client to schedule when there are multiple clients with the same largest $r_n(t)$. Let $X_\pi(t)$ denote the random process of $X(t)$ under a scheduling policy $\pi \in \Pi^*$. Then, given any scheduling policy $\pi \in \Pi^*$,

$$X_\pi(t) \geq X_\eta(t), \quad \forall t \geq 0, \tag{3.41}$$

for every sample path, for any scheduling policy η. Let $\Delta X_\pi(t+1) := X_\pi(t+1) - X_\pi(t)$. Since $R(t)$ is i.i.d. across all time slots, $\Delta X_\pi(t)$ is also i.i.d. for all $t > 0$. Moreover,

$$E[\Delta X_\pi(t)] = E[R(t)] - \sum_{n=1}^{N} q_n = 0 \tag{3.42}$$

$$\mathrm{Var}[\Delta X_\pi(t)] = \mathrm{Var}[R(t)] = E[(R(t))^2] - \left(\sum_{n=1}^{N} q_n\right)^2 \tag{3.43}$$

Then, by the functional central limit theorem for i.i.d. random variables [19], we summarize the fundamental properties of the diffusion limit of $X_\pi(t)$ as follows.

Theorem 15. *For any scheduling policy $\pi \in \Pi^*$, define $\hat{X}_\pi(t) := \lim_{k\to\infty} \frac{X_\pi(kt)}{\sqrt{k}}$ and $\sigma^2 := E[(R(t))^2] - \left(\sum_{n=1}^N q_n\right)^2$. Then, $\hat{X}_\pi(t)$ is a driftless Brownian motion with variance σ^2. Furthermore, given $\hat{X}_\pi(\tau)$, for any $\tau, t \geq 0$ with $\tau + t \leq 1$, $\hat{X}_\pi(\tau + t) - \hat{X}_\pi(\tau)$ is a Gaussian random variable with zero mean and variance $\sigma^2 t$.* $\square$

By Theorem 15, we know that $\hat{X}_\pi(t)$ has the same behavior for all scheduling policy $\pi \in \Pi^*$. For simplicity, we use $\hat{X}^*(t)$ to denote the diffusion limit of $X_\pi(t)$ for any policy $\pi \in \Pi^*$.

3.5.2 Variable-Bit-Rate Videos

Next, we study the dynamics of variable-bit-rate videos to characterize the behavior of $\hat{Y}_n(t)$. Define

$$Z_n(t) := C_n(\lfloor \frac{t}{k_n} \rfloor) - q_n t. \tag{3.44}$$

Note that $C_n(\lfloor \frac{t}{k_n} \rfloor)$ is the total number of frames played up to t if $D_n(t) = 0$. Similar to $Y_n(t)$, $Z_n(t)$ aims to capture the dynamics of video frame size but without taking video interruption into account. We consider $Z_n(t)$ for two reasons:

- $Z_n(t)$ and $Y_n(t)$ behave the same in the diffusion limit. This will be shown later in Lemma 9.

- Using $Z_n(t)$ instead of $Y_n(t)$ greatly simplifies the design and implementation of scheduling policy. This will become more clear in Section 3.5.4 (see Remark 13).

Consider the diffusion limits of $Z_n(t)$ and $C_n(t)$ as $\hat{Z}_n(t) := \lim_{k\to\infty} \frac{Z_n(kt)}{\sqrt{k}}$ and $\hat{C}_n(t) := \lim_{k\to\infty} \frac{C_n(kt) - q_n^* kt}{\sqrt{k}}$. We state a useful property in the following lemma.

Lemma 9. *For any client n, at any t, we have*

$$\hat{Y}_n(t) = \hat{Z}_n(t) = \hat{C}_n\Big(\frac{t}{k_n}\Big), \tag{3.45}$$

where $\hat{C}_n(t)$ is a driftless Brownian motion with variance $\sigma^2_{q,n}$. Therefore, both $\hat{Z}_n(t)$ and $\hat{Y}_n(t)$ are driftless Brownian motion with variance $\sigma^2_{z,n} = \frac{\sigma^2_{q,n}}{k_n}$, regardless of scheduling policy. □

Proof. The proof is provided in Appendix B.3. □ □

By Lemma 9, we are able to fully characterize the distribution of $\hat{Y}_n(t)$ and $\hat{Z}_n(t)$, regardless of the scheduling policy.

3.5.3 A Lower Bound of Capacity Region

By Lemma 9, (3.19) can be written as $\hat{D}_n(t) = \sup_{0\le\tau\le t}(\max\{0, -\frac{\hat{X}_n(t)-\hat{Z}_n(t)}{q_n}\})$. Define

$$Z(t) := \sum_{n=1}^{N} Z_n(t) = \sum_{n=1}^{N} C_n(\lfloor\frac{t}{k_n}\rfloor) - \sum_{n=1}^{N} q_n t. \tag{3.46}$$

Since $\hat{Z}_n(t)$ is a driftless Brownian motion and $\hat{Z}_n(t)$ are independent among different n, then $\hat{Z}(t) := \lim_{k\to\infty} \frac{Z(kt)}{\sqrt{k}}$ is also a driftless Brownian motion with variance $\sigma_z^2 := \sum_{n=1}^{N} \sigma^2_{z,n}$. Similar to (3.27), we define

$$\hat{D}^*(t) := \sup_{0\le\tau\le t} (\max\{0, -(\hat{X}^*(\tau) - \hat{Z}(\tau))\}). \tag{3.47}$$

Note that $\hat{X}^*(\tau) + (-\hat{Z}(\tau))$ is the sum of two independent driftless Brownian motion and therefore is also a driftless Brownian motion with variance $(\sigma^2 + \sigma_z^2)$. By using

a similar argument as (3.29)-(3.31), we further have

$$\hat{D}^*(t) \leq_{st} \sum_{n=1}^{N} q_n \hat{D}_n(t), \tag{3.48}$$

under any such scheduling policy.

Definition 11. *For a system with fading channels and variable-bit-rate videos, a vector* $\lambda = [\lambda_1, \lambda_2, ..., \lambda_N]$ *is said to be feasible if there exists a scheduling policy such that*

$$\hat{D}_n(t) \leq_{st} \frac{\lambda_n}{q_n} \hat{D}^*(t), \quad n = 1, 2, ..., N. \tag{3.49}$$

Then, the capacity region for QoE is defined as the set of all feasible vectors λ. □

Again, we obtain a lower bound of capacity region as follows.

Theorem 16. *For a system with fading channels and variable-bit-rate videos, a feasible vector* $\lambda = [\lambda_1, \lambda_2, ..., \lambda_N]$ *with* $\lambda_n \geq 0$*, for all* n*, must satisfy* $\sum_{n=1}^{N} \lambda_n \geq 1$. □

Proof. Similar to the proof of Theorem 11, this can be proved by using (3.48) and Definition 11. □ □

3.5.4 Scheduling Policy

For fading channels and variable-bit-rate videos, we propose the following extended version of the JCD policy.

Highest Data Rate Policy For Variable-Bit-Rate Videos (HDR-VBR):

In each time slot t, the AP schedules a client with the largest $r_n(t)$ and break ties by choosing the one with the smallest $w_n(X_n(t) - Z_n(t))$, where w_n is a predetermined weight factor. □

Remark 12. Note that $X_n(t) - Z_n(t) = (A_n(t) - q_n t) - (C_n(\lfloor \frac{t}{k_n} \rfloor)) - q_n t) = A_n(t) - C_n(\lfloor \frac{t}{k_n} \rfloor)$, which reflects the difference between the total received video content and the total video content that should have been played if there is no video interruption at all. Hence, $X_n(t) - Z_n(t)$ still loosely reflects the status of the playback buffer of client n.

Remark 13. Under the HDR-VBR policy, the AP requires the information of $A_n(t)$ and $C_n(\lfloor \frac{t}{k_n} \rfloor)$. In wireless networks, $A_n(t)$ can be obtained by collecting ACKs from the clients. For $C_n(\lfloor \frac{t}{k_n} \rfloor)$, since the AP has the video files, the AP can simply refer to the accumulative size of the frames that should have been played up to current time t. Hence, the HDR-VBR policy can be easily implemented on the AP.

Remark 14. For the special case of *constant-bit-rate* videos, the HDR-VBR policy degenerates to the *HDR policy* studied in [25]. Under the HDR policy, the AP schedules a client with the largest $r_n(t)$ and break ties by choosing the one with the smallest $w_n X_n(t)$ in each time slot t.

Theorem 17. *Let w_n be the predetermined weight for client n. For variable-bit-rate video, under the HDR-VBR policy and conditions (3.37) and (3.38), we have $w_n(\hat{X}_n(t) - \hat{Z}_n(t)) = w_m(\hat{X}_m(t) - \hat{Z}_m(t))$, for any pair of clients n, m.* □

Proof. The proof is provided in Appendix B.4. □ □

Given the state-space collapse property, the HDR-VBR can achieve every interior point in the capacity region. The key results are summarized in the following theorem.

Theorem 18. *Given any vector* $\lambda = [\lambda_1, \lambda_2, ..., \lambda_N]$ *which satisfies* $\lambda_n > 0$, $\forall n$, *and* $\sum_{n=1}^{N} \lambda_n \geq 1$, *HDR-VBR policy can achieve* $\hat{D}_n(t) = \frac{\frac{1}{w_n}}{q_n \sum_{m=1}^{N} \frac{1}{w_m}} \hat{D}^*(t) \leq_{st} \frac{\lambda_n}{q_n} \hat{D}^{**}(t)$ *by assigning* $w_n = \frac{1}{\lambda_n}$ *for all* n. *Moreover, we have*

$$E[\hat{D}_n(t)] = \sqrt{\frac{2t(\sigma^2 + \sigma_z^2)}{\pi}} \frac{\frac{1}{q_n w_n}}{\sum_{m=1}^{N} \frac{1}{w_m}}. \tag{3.50}$$

Remark 15. In (3.50), we see that video interruption arises from two factors: σ^2 due to the randomness in fading channels and σ_z^2 due to the randomness in variable-bit-rate videos.

Based on Theorems 16 and 18, we characterize the capacity region for fading channels and variable-bit-rate videos.

Theorem 19. *For a system with fading channels and variable-bit-rate videos, a vector* $[\lambda_1, \lambda_2, \dots]$ *with* $\lambda_n > 0$, $\forall n$ *is feasible if and only if* $\sum_{n=1}^{N} \lambda_n \geq 1$. □

3.6 Network Utility Maximization for QoE

In this section, we propose a network utility maximization (NUM) problem for QoE, and obtain tractable solutions for special cases. Given $\hat{D}_n(t)$ and T, we assume that each client n suffers from some *penalty* $f_n(E[\hat{D}_n(T)])$, where $f_n(\cdot)$ is an increasing, differentiable, and convex function. Note that the expectation $E[\hat{D}_n(T)]$ is taken over all sample paths for $t \in [0, T]$. Here we use $E[\hat{D}_n(T)]$ to approximate the short-term playback interruption $D_n(T)$. We then aim to minimize the total penalty in the system, which can be expressed as $\sum_n f_n(E[\hat{D}_n(T)])$.

By Theorems 14 and 19, we have $\sum_n q_n E[\hat{D}_n(T)] \geq E[\hat{D}(T)]$ for ON-OFF channels and $\sum_n q_n E[\hat{D}_n(T)] \geq E[\hat{D}^*(T)]$ for fading channels and variable-bit-rate videos.

Further, if we have the additional condition of $E[\hat{D}_n(T)] > 0$, the JCD policy and the HDR-VBR policy can achieve any set of $\{E[\hat{D}_n(T)]\}$ by properly assigning the weight $\{w_n\}$ to each client. Since the formulation of the NUM problem is the same for ON-OFF channels and fading channels plus variable-bit-rate videos, we consider the more general case in the rest of this section.

Below, we study an example of NUM problem which aims to minimize the sum of polynomial penalty functions.

NUM with Polynomial Penalty Functions:

Given the distribution of $\mathbf{r}(t)$, $T > 0$, $\alpha \geq 1$, and a vector $[\delta_1, \delta_2, \dots]$ with $\delta_n > 0$, for all n,

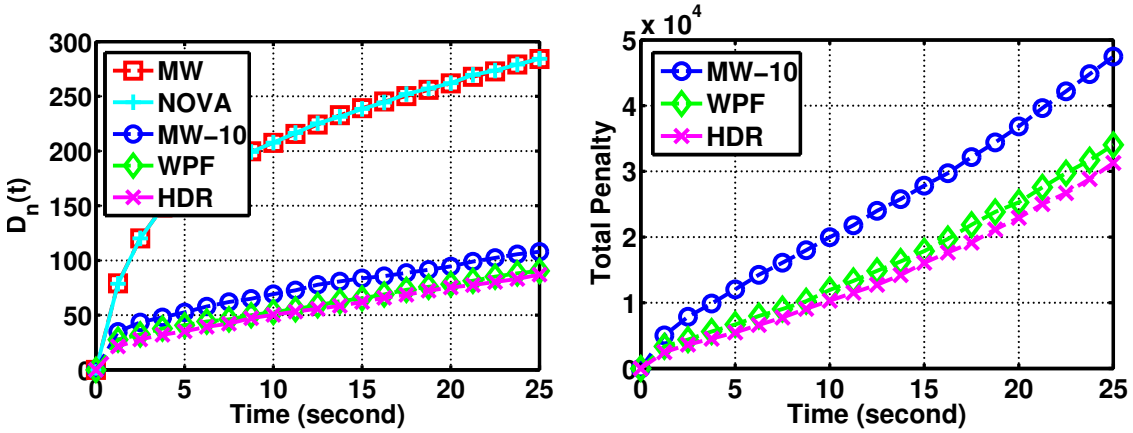
$$\text{Min.} \quad \sum_{n=1}^{N} \delta_n \cdot (E[\hat{D}_n(T)])^{\alpha}$$

$$\text{s.t.} \quad q_1 E[\hat{D}_1(T)] + \cdots + q_N E[\hat{D}_N(T)] \geq E[\hat{D}^*(T)]. \quad \square$$

To minimize the total penalty, we define a function L_1 with a Lagrange multiplier μ_1 as

$$L_1 = \sum_{n=1}^{N} \delta_n (E[\hat{D}_n(T)])^{\alpha} - \mu_1 \Big(\sum_{n=1}^{N} q_n E[\hat{D}_n(T)] - E[\hat{D}^*(T)] \Big).$$

Next, we take the partial derivative of L_1 with respect to each $E[\hat{D}_n(T)]$ and set them to zero, i.e. $\frac{\partial L_1}{\partial E[\hat{D}_n(T)]} = \delta_n \alpha (E[\hat{D}_n(T)])^{\alpha-1} - \mu_1 q_n = 0$, $\forall n$. If $\alpha > 1$, an optimal solution occurs when $\frac{E[\hat{D}_n(T)]}{E[\hat{D}_m(T)]} = \left(\frac{q_n/\delta_n}{q_m/\delta_m} \right)^{\frac{1}{\alpha-1}}$, for any pair n, m. From (3.50), it is equivalent to have $\frac{\beta_n}{\beta_m} = \frac{q_n^{\frac{\alpha}{\alpha-1}} \cdot \delta_n^{\frac{-1}{\alpha-1}}}{q_m^{\frac{\alpha}{\alpha-1}} \cdot \delta_m^{\frac{-1}{\alpha-1}}}$. Then, we can simply assign $w_n = \delta_n^{\frac{1}{\alpha-1}} q_n^{\frac{-\alpha}{\alpha-1}}$ for each client so that HDR-VBR achieves an optimal solution. If $\alpha = 1$, the problem



(a) Average duration of video interruption. (b) Total penalty of the network.

Figure 3.1: Comparisons of the five policies in a fully symmetric system with constant-bit-rate videos.

degenerates to a linear program. An optimal solution is obtained by assigning almost all the video interruption time to a client with the smallest $\frac{\delta_n}{q_n}$. Without loss of generality, we may assume that $\frac{\delta_1}{q_1} \leq \frac{\delta_2}{q_2} \leq ... \leq \frac{\delta_N}{q_N}$. Then, we just assign $w_1 = 1$ and let w_n be extremely large for the other clients.

3.7 Simulation Results

We evaluate the proposed policies through ns-2 simulation. Following the IEEE 802.11a standard, we simulate a wireless network that allows data transmission at 54, 48, 36, 18, and 6 Mbit/s. The time to transmit a packet and to receive an ACK is set to be 500 μs, which is short enough so that the channel quality stays almost the same in a time slot. We thereby obtain the corresponding packet size for each data rate: 2340, 2080, 1560, 750, 220 bytes. The frame rate of each video stream is 30 frames per second, and thus each client plays one frame every 33.3 milliseconds (equivalent to about 66 time slots). All the results presented in this section are the average of 50 simulation trials. We compare HDR-VBR policy against four policies: HDR policy,

Max-Weight policy (MW), weighted Proportional-Fair policy (WPF), and NOVA algorithm. In MW policy, the AP schedules the one with the largest $(-r_n X_n(t))$ and breaks tie by choosing the one with the largest $(-X_n(t))$. To further explore the difference between HDR-VBR and MW, we also consider the Max-Weight-α policy (MW-α), which schedules the client with the largest $r_n(\max(0, -X_n(t)))^{\frac{1}{\alpha}}$. When $\alpha > 1$, the instantaneous data rate becomes more influential than $X_n(t)$. In the following simulations, we assign $\alpha = 10$. For WPF policy, the scheduled client at time t is the one that maximizes $q_n(r_n(t)/A_n(t-1))$ [54]. For NOVA, we choose the same objective function as that in [46] with a slight change in the initial condition (b_{i0} in [46]) to fit in our simulation scenario.

3.7.1 Constant-Bit-Rate Videos

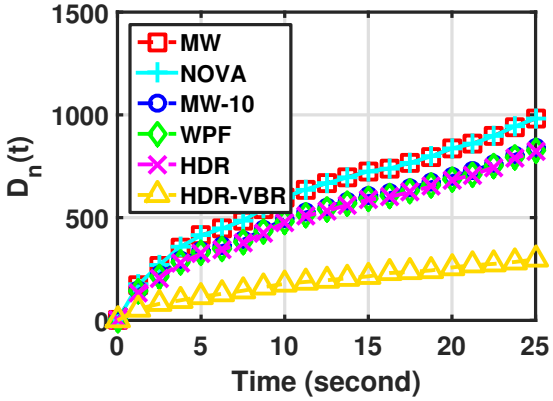
For the special case of constant-bit-rate videos, HDR-VBR is exactly the same as the HDR policy. Therefore, we merge the results of HDR and HDR-VBR for constant-bit-rate videos. We consider a fully symmetric system of 20 clients under the heavy-traffic condition given by (3.37) and (3.38). The channel distribution of each client is the same and evenly distributed, i.e. the probability of each data rate is 0.2. Under this setting, $E[R(t)] \approx 2340$. Therefore, q_n is chosen to be 117 byte/slot for every client. We study a quadratic QoE objective function given by $\sum_n \delta_n (E[\hat{D}_n(T)])^2$, where $\delta_n = 1$ for all n. Since the system is fully-symmetric, we choose $w_n = 1$ for all the clients. Fig. 3.1 shows the results of the symmetric system. In Fig. 3.1(a), HDR has the smallest $D_n(t)$ among all the policies, while MW and NOVA perform rather poorly. As expected, MW-10 policy has a moderate $D_n(t)$ since MW-10 serves as an intermediate between MW and HDR. Moreover, it is noticeable that WPF has similar performance to HDR. The main reason is that in the symmetric case, $A_n(t)$ of each client grows almost at the same speed and

thus maximizing $q_n(r_n(t)/A_n(t-1))$ is equivalent to maximizing $r_n(t)$ in each slot. Moreover, Fig. 3.1(b) shows the total penalty of MW-10, WPF, and HDR policy to further compare the difference between these three policies.

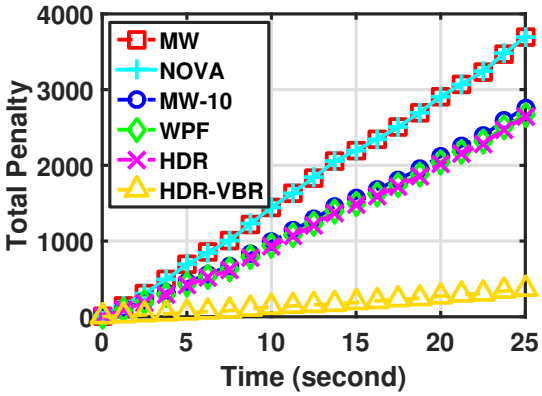
3.7.2 Variable-Bit-Rate Videos

Following the same simulation setup as that of constant-bit-rate videos, we evaluate the HDR-VBR policy against HDR as well as other popular policies with variable-bit-rate videos. First, we consider a fully-symmetric system as that in Section 3.7.1 but with variable-bit-rate videos. We assume that the frame size of each video is uniformly distributed between 100 bytes and 15344 bytes with average frame size of 7722 bytes. This corresponds to an average playback rate of 117 bytes per slot for every client. Fig. 3.2(a) shows the average playback interruption of the fully-symmetric system. As expected, the HDR-VBR has the smallest $D_n(t)$ among all the policies. More importantly, with variable-bit-rate videos, the HDR-VBR policy outperforms the HDR policy since the original HDR does not include the information about variable playback rates. Besides, Fig. 3.2(b) demonstrates the total penalty incurred by playback interruption. It is also noticeable that the total penalty is larger in Fig. 3.2(b) than that in Fig. 3.1(b) due to the fluctuation in video playback rates.

Next, we turn to the asymmetric case. We divide the clients equally into two classes. We assign $\delta_n = 10$ to Class 1 and $\delta_n = 1$ to Class 2, with the result that Class 1 dominates the overall QoE performance. In addition, we assume that the two classes have the same evenly-distributed channel but different playback rates. Suppose the clients in Class 1 and Class 2 watch videos with resolution of 480p and 720p, respectively. According to the recommended bitrates for YouTube videos in [55], we choose $q_n = 156$ and 78 bytes/slot for 720p and 480p videos, respectively. To include randomness in frame sizes, we assume that each client in Class 1 plays

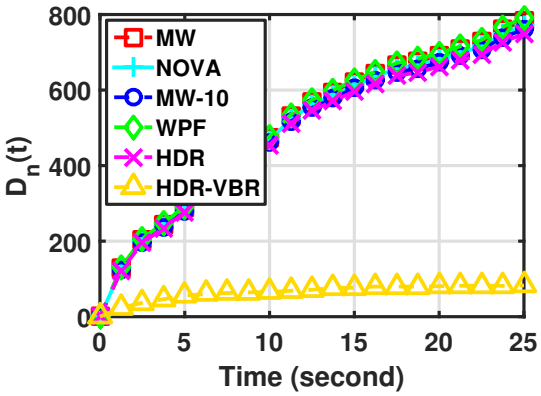


(a) Average duration of video interruption.

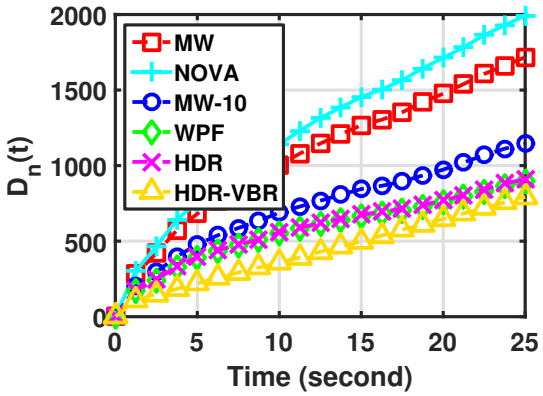


(b) Total penalty of the network.

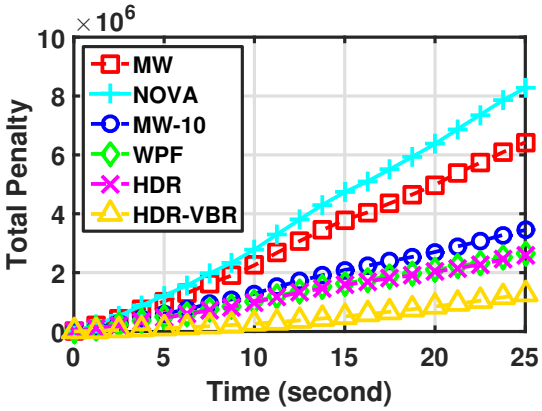
Figure 3.2: Comparisons of the six policies in a fully symmetric system with variable-bit-rate videos.



(a) Average playback interruption: Class 1.



(b) Average playback interruption: Class 2.



(c) Total penalty of the network.

Figure 3.3: Comparisons of the five policies in a system with the same channel distribution but heterogeneous playback rates.

Table 3.2: Information of the Videos in the Experiments.

Video #	Video Name	Avg. Bitrate (Mbps)	Frame Rate
1	The Simpsons (Official Trailer)	2.04	24
2	Serenity (Official Trailer)	2.25	24
3	Toy Story 3 (Official Trailer)	0.71	30
4	Angry Birds (Official Trailer)	0.65	24
5	The Simpsons (Official Trailer, HD)	3.42	24

a video with frame size uniformly distributed between a minimum 100 bytes and maximum 10196 bytes with average playback rate of 78 bytes per slot. Similarly, each client in Class 2 plays a video with frame size uniformly distributed between 100 bytes and 20492 bytes with average playback rate of 156 bytes per slot. By the discussion in Section 3.6, we assign $w_n = 40$ to Class 1 and $w_n = 1$ to Class 2 to optimize the network utility under the HDR-VBR policy. Fig. 3.3 shows the playback interruption and the total penalty of the asymmetric system. Clearly, the HDR-VBR still achieves the smallest playback interruption for both clients in Class 1 and Class 2 by taking the fluctuation in video frame size into account. Moreover, the HDR-VBR policy intelligently allocates $D_n(t)$ among the two classes by assigning proper weights w_n so that it can achieve the smallest total penalty.

From simulation, we note that all of the five policies are stabilizing since the duration of video interruption grows sublinearly. However, the short-term performance of these policies are rather different in the heavy-traffic regime. Therefore, diffusion limit indeed provides more detailed information on the playback process.

3.8 Experimental Results With Real Videos

We further evaluate the performance of the proposed policies with real videos on a software-defined wireless testbed. The experiments are done on PULS, which

Table 3.3: Experimental Results Under Heavy-Traffic Condition.

Policy	Trial	$D_1(t)$ (sec)	$D_2(t)$ (sec)	$D_3(t)$ (sec)	$D_4(t)$ (sec)	$D_5(t)$ (sec)	RX Thruput (Mbps)
HDR-VBR	#1	3.12	9.14	4.06	4.11	0.00	8.39
	#2	3.41	9.29	4.27	4.09	0.00	8.40
	#3	2.72	9.01	3.80	3.46	0.00	8.55
	#4	2.36	7.48	3.88	3.83	0.00	8.57
	#5	0.29	7.32	1.24	1.06	0.00	9.47
	#6	0.19	7.33	1.21	1.02	0.00	9.36
WPF	#1	9.78	9.61	10.36	8.96	0.00	8.47
	#2	9.36	9.46	9.99	8.62	0.00	8.51
	#3	9.26	9.38	9.85	8.46	0.00	8.59
	#4	6.56	7.59	6.77	5.21	0.00	9.46
	#5	8.33	8.59	8.61	7.54	0.00	8.85
	#6	7.10	8.17	7.30	6.00	0.00	9.19

Table 3.4: Experimental Results Under Non-Heavy-Traffic Condition.

Policy	Trial	$D_1(t)$ (sec)	$D_2(t)$ (sec)	$D_3(t)$ (sec)	$D_4(t)$ (sec)	$D_5(t)$ (sec)	RX Thruput (Mbps)
HDR-VBR	#1	0.00	4.02	0.00	0.00	0.00	11.93
	#2	0.00	3.59	0.00	0.00	0.00	11.98
	#3	0.00	3.56	0.00	0.00	0.00	12.30
	#4	0.00	3.48	0.00	0.00	0.00	12.43
	#5	0.00	3.64	0.00	0.00	0.00	12.38
	#6	0.00	3.53	0.00	0.00	0.00	12.44
WPF	#1	2.54	4.00	1.42	1.57	0.00	11.88
	#2	1.99	3.37	0.72	1.03	0.00	12.62
	#3	2.03	3.41	0.78	1.08	0.00	12.55
	#4	2.15	3.50	0.91	1.22	0.00	12.48
	#5	1.97	3.37	0.71	1.03	0.00	12.54
	#6	2.07	3.43	0.81	1.12	0.00	12.41

is a FPGA-based wireless testbed presented in [56, 57]. With the clean separation between software and hardware, PULS allows us to quickly prototype the proposed

scheduling policies completely in the software domain. More details of the testbed will be introduced in Chapter 5.

We consider five on-demand videos streams from an AP to five different clients using real video files. Table 3.2 shows the basic information about the videos played in the experiments. For simplicity, we consider symmetric wireless channels for all the clients by having the five clients co-located at the same station. For the MAC and PHY specification, the AP and the clients follow the IEEE 802.11a standard. Suppose the AP aims to minimize a linear objective function as $\sum_n \delta_n E[D_n(t)]$ with $\delta_1 = \delta_5 = 10$ and $\delta_2 = \delta_3 = \delta_4 = 1$. This implies that client 1 and client 5 should expect better QoE than the other three clients. By the results in Section 3.6, since client 2 has the smallest $\frac{\delta_n}{q_n}$, we choose $w_n = 1$ for client 2 and $w_n = 1000$ for the rest of the clients. We compare the HDR-VBR policy with the WPF policy, which has already been shown to have better performance than MW and NOVA policy in simulations.

3.8.1 Heavy-Traffic Case

We first run experiments under heavy-traffic condition, i.e. the total receiver throughput is about the same as total video playback rate. By using a fixed 16-QAM modulation and coding rate 1/2, the total receiver throughput is about 9 Mbps. Figure 3.4 shows the experimental results under the HDR-VBR and WPF policy. Compared to the WPF policy, the HDR-VBR indeed significantly reduces playback interruption by keeping track of the variation in video bit rates. Moreover, Table 3.3 provides the accumulated playback interruption at 40 second and the average receiver throughput of each experiment trial. From Table 3.3, we know that HDR-VBR performs much better than WPF while the receiver throughput under both policies are almost the same.

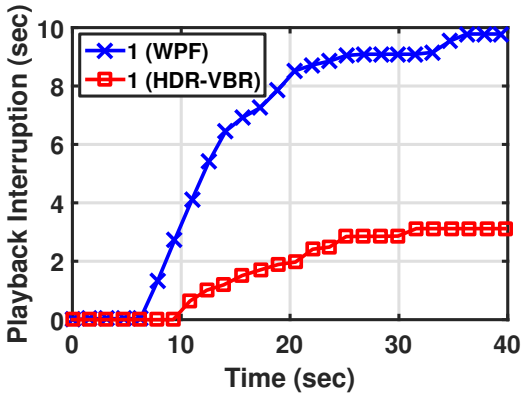
3.8.2 Non-Heavy-Traffic Case

We increase the receiver throughput by using 64-QAM modulation and coding rate 3/4. Under this setting, the receiver throughput is about 12 Mbps, which corresponds to roughly 75% system load. Figure 3.5 shows the video interruption of each client and the total penalty under the HDR-VBR and WPF policy. Due to the increase in throughput, both policies achieve much less video interruption than in the heavy-traffic case. In this case, HDR-VBR achieves zero video interruption for client 1, 3, 4, and 5 while still maintaining about the same amount of video interruption for client 2 as that under WPF. Table 3.4 further verifies this result through multiple experimental trials.

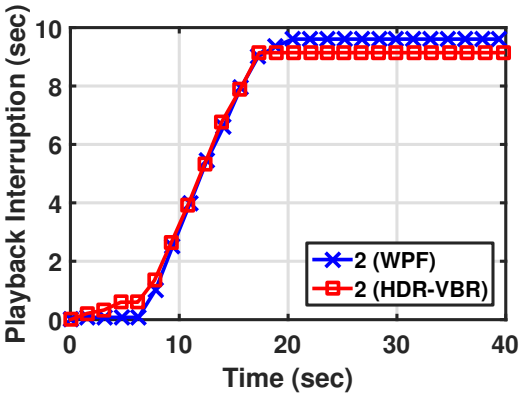
Hence, the experiments demonstrate that HDR-VBR outperforms its counterparts with real videos under both heavy-traffic and non-heavy-traffic conditions.

3.9 Summary

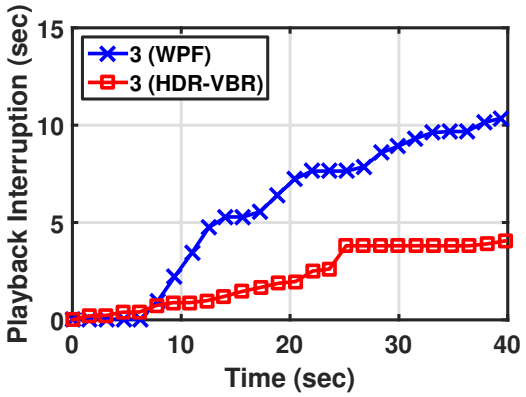
In this chapter, we study dynamic behavior of QoE in heavy traffic by using diffusion approximation. We characterize the capacity region for QoE and propose online scheduling policies to optimize QoE. Simulation and experimental results show that the proposed policies outperform existing popular policies. In the future, we intend to further study the effect of adaptive video bit rates and user engagement on QoE.



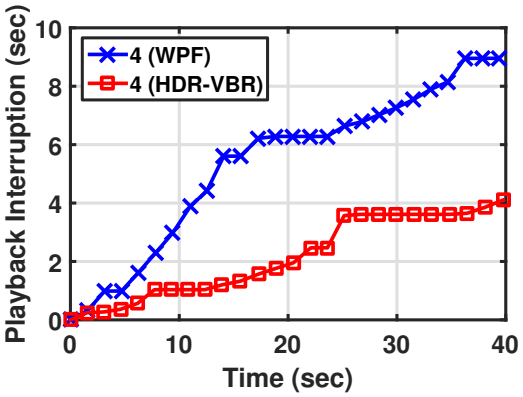
(a) Playback interruption: Client 1.



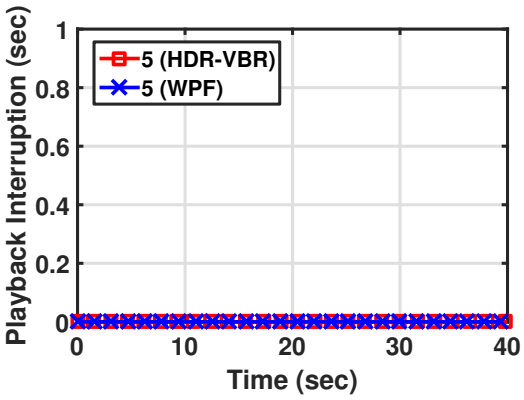
(b) Playback interruption: Client 2.



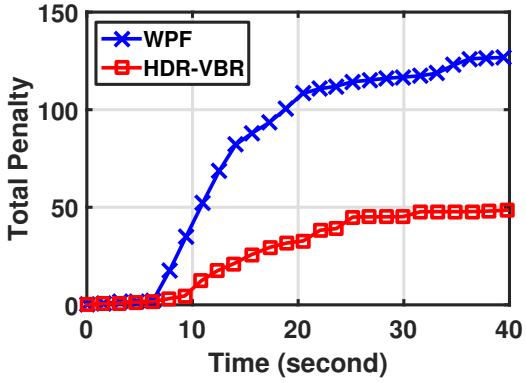
(c) Playback interruption: Client 3.



(d) Playback interruption: Client 4.

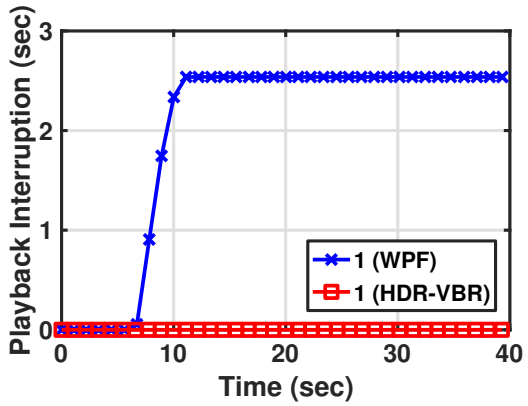


(e) Playback interruption: Client 5.

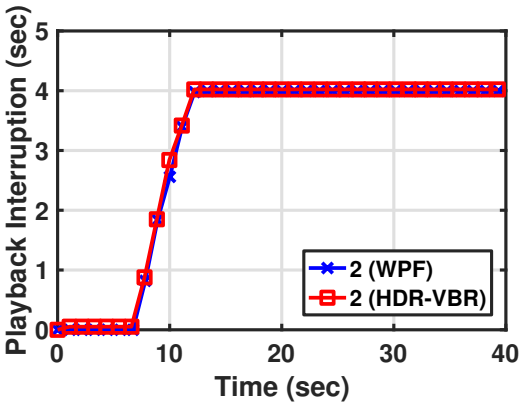


(f) Total penalty of the network.

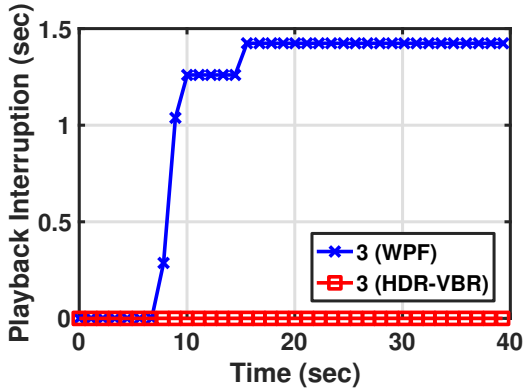
Figure 3.4: Experimental results with five real video streams under heavy-traffic condition.



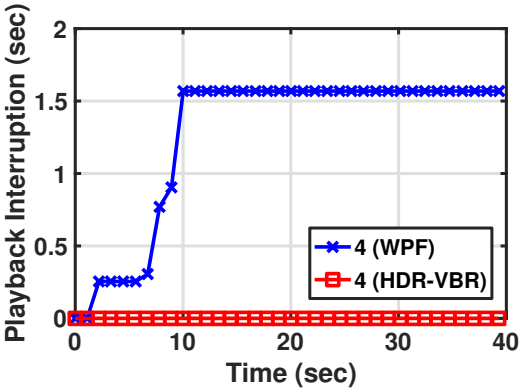
(a) Playback interruption: Client 1.



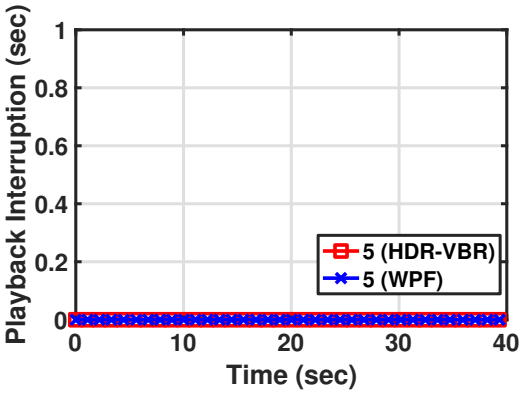
(b) Playback interruption: Client 2.



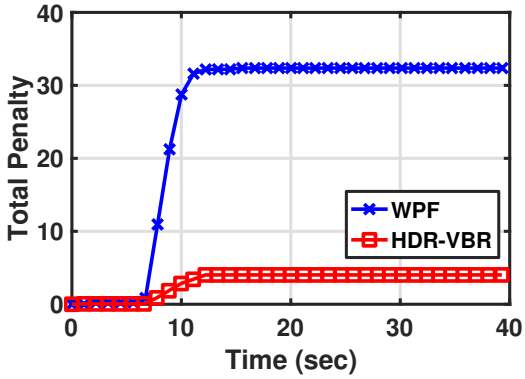
(c) Playback interruption: Client 3.



(d) Playback interruption: Client 4.



(e) Playback interruption: Client 5.



(f) Total penalty of the network.

Figure 3.5: Experimental results with five real video streams under non-heavy-traffic condition.

4. A DECENTRALIZED PROTOCOL FOR REAL-TIME WIRELESS AD HOC NETWORKS[1]

This chapter describes the proposed research for designing distributed protocols for real-time wireless networks for IoT applications, such as networked industrial control systems.

4.1 Introduction

Real-time wireless networks are becoming essential to a variety of existing and emerging applications. For multimedia applications such as virtual reality and live video streaming, video contents need to be delivered from the content providers to the end users within several tens of milliseconds to provide seamless user experience. On the other hand, cyber-physical systems (CPS), such as industrial Internet of Things (IoT) applications and networked transportation systems, usually require ultra-low per-packet latency within several milliseconds as well as very low packet loss rate to achieve reliable real-time control. These characteristics are usually captured by *timely-throughput*, which is defined as the average throughput of on-time packet deliveries.

To achieve the timely-throughput requirements, researchers have been devoting significant efforts to designing scheduling algorithms with provable performance guarantees. In [59], Dua and Bambos study downlink scheduling for packets with deadlines by applying dynamic programming. In [60], Hou et al. propose a mathematical framework for designing centralized online scheduling algorithms for deadline-constrained wireless networks over static unreliable channels. This framework has

[1]Part of this chapter is reprinted with permission from "A Decentralized Medium Access Protocol for Real-Time Wireless Ad Hoc Networks With Unreliable Transmissions" by Ping-Chun Hsieh and I-Hong Hou in Proc. of IEEE ICDCS 2018 [58].

been extended to various wireless scenarios, such as fading channels [61], multicast traffic [62], broadcast traffic [63], coexistence of real-time and non-real-time traffic [64], arbitrary traffic patterns [65], and wireless ad hoc networks [66, 67]. These algorithms make scheduling decisions based on the state information of each wireless link, such as virtual queue length and the number of packet arrivals. While these scheduling algorithms have provably good performance, they need to maintain state information at the centralized controller (such as a Wi-Fi access point (AP) or a cellular base station) and might not be practical in certain settings. For example, when there are multiple collocated access points sharing the same wireless medium, either for better coverage or higher network density, it is required to incorporate additional coordination between the access points in order to apply these centralized scheduling algorithms. Such coordination can be extremely difficult for real-time wireless networks due to stringent per-packet deadlines. Moreover, there might be direct device-to-device communication without the help of an AP (such as ad hoc mode of IEEE 802.11 and WiFi Direct [68]) for message exchange between sensors and actuators [69]. Furthermore, to apply centralized control in a wireless network with uplink traffic, an access point needs to continually update global information through polling. This may incur prohibitively large scheduling overhead, especially when the network size is large.

In light of the above issues of centralized scheduling, recently researchers have embarked on designing random-access algorithms to allow uncoordinated devices to share the same wireless medium efficiently [70, 71, 72, 73, 74, 75, 76, 77, 78, 79, 80, 81]. The existing studies on random-access algorithms can be divided into two main categories:

1. **Providing throughput guarantees for *non-real-time* wireless networks**:

To achieve throughput-optimality in wireless ad hoc networks, Jiang and Walrand [70] and Rajagopalan, Shah, and Shin [71] propose two different CSMA-based distributed algorithms by using queue-length-based continuous backoff scheme under the assumption of perfect carrier sensing. The results are later extended for utility maximization with congestion control [72, 73], circuit-switched networks [74], and various weight functions for backoff timers [75]. Without assuming perfect carrier sensing, Ni et al. [76] develop a throughput-optimal queue-length-based CSMA algorithm for wireless ad hoc networks by using discrete backoff timers and control packets. To overcome the limitations of both perfect carrier sensing and control packets, Shah et al. [77] propose a throughput-optimal random access algorithm by intelligently choosing a weight function that depends only on the local queue and sensing information. Lotfinezhad and Marbach [78] propose a CSMA-based algorithm that achieves throughput-optimality as well as order-optimal delay for wireless ad hoc networks, and Paolini et al. [79] propose a coded random access scheme based on slotted ALOHA to support massive uncoordinated wireless access. The above list is by no means exhaustive, but it provides a portrait of the recent progress in random access algorithms for non-real-time wireless networks. Despite the significant progress, none of the above algorithms takes per-packet deadlines into consideration.

2. **Providing timely-throughput guarantees for *real-time* wireless networks**: To accommodate per-packet deadlines, based on a similar approach as [70], Li and Eryilmaz [80] propose the FCSMA algorithm, a CSMA-based distributed implementation of [66] for a fully-connected network. While FCSMA algorithm has been shown to be feasibility-optimal, the optimality results are based on the assumption that there is no capacity loss due to backoff overhead or collisions. For

real-time wireless networks with stringent packet deadlines, backoff overhead can lead to significant capacity loss and needs to be taken into account. Moreover, one common issue of CSMA with random backoff is that collision rate increases with the network size. It has been shown analytically that the collision probability for the Distributed Coordination Function (DCF) of Wi-Fi protocols can be prohibitively high, and that the throughput loss due to collision is significant, under the exponential backoff scheme even when the network size is fairly small (e.g. 10 links) [82]. On the other hand, by following a similar approach as [76], Lu et al. [81] present the frame-based CSMA algorithm, which distributedly generates schedules on a per-frame basis by incorporating control packets and a control slot at the beginning of each frame. While the frame-based CSMA algorithm has been shown to be feasibility-optimal for wireless ad hoc networks with reliable transmissions, it is sub-optimal with unreliable channels since the schedules are not adaptive to the packet losses within a frame.

In this research, we aim to *achieve optimal timely-throughput over unreliable channels in a decentralized manner while avoiding the capacity loss due to channel contention (including collision and backoff overhead).* We are particularly interested in real-time wireless ad hoc networks for industrial control systems, where sensors and actuators exchange critical control messages via multiple APs and direct device-to-device communication. Given the limited distance between devices in the same manufacturing area, each device is likely to cause severe interference to all the other devices. In this regard, we propose decentralized priority-based algorithms for fully-interfering networks to optimize network timely-throughput and address the channel contention issue simultaneously.

Different from conventional CSMA-based algorithms, the proposed algorithm let

all the links contend for *transmission priorities* in addition to immediate channel access. Instead of using random backoff, which is commonly used by CSMA-based algorithms, we utilize a collision-free backoff scheme as the tool to determine transmission priorities of all the links. In this way, the proposed algorithm is free from collisions and have quantifiably small backoff overhead. Based on the transmission priorities, the schedules can automatically adapt to packet losses in a fully decentralized fashion. Moreover, by dynamically adjusting the transmission priorities in a randomized manner, the proposed decentralized algorithm achieves optimal timely-throughput as its centralized counterparts.

It is important to note that conventional CSMA-based algorithms usually suffer from slow convergence and large delay due to the "locking" effect [78, 83]. Specifically, CSMA-based algorithms tend to stick to one schedule for a long time and hence result in starvation of the other unscheduled links. To tackle this issue, there have been significant efforts in improving delay performance and convergence time of CSMA algorithms, such as [78, 83, 76, 84, 85, 86, 87, 88, 89]. By contrast, one advantage of the proposed priority-based algorithm is that the priority structure mitigates the locking effect by nature. Specifically, under any priority ordering, each link receives a non-zero expected timely-throughput depending on its priority index. Therefore, even the link with the lowest priority does not get completely starved. This design shares a similar philosophy as the virtual multi-channel approach proposed by [83]. More details on convergence time will be discussed via simulation in Section 4.6.

The main contributions can be summarized as follows:

- We propose a generic fully-decentralized priority-based protocol for wireless networks. The proposed protocol requires only carrier sensing and backoff mechanism and does not require any additional control packets or control slots.

- The proposed protocol utilizes a collision-free backoff mechanism and therefore completely avoids capacity loss due to collided transmissions. Meanwhile, the overhead due to backoff is quantifiably small compared to the packet deadlines. The proposed protocol is robust to packet losses resulting from unreliable transmissions.

- For wireless networks with per-packet deadlines, by combining the proposed protocol with virtual queue lengths, we propose a fully-decentralized priority-based algorithm that is feasibility-optimal.

- We evaluate the proposed algorithm via extensive NS-3 simulations and show that it indeed performs as well as the optimal centralized algorithms.

The rest of this chapter is organized as follows. Section 4.2 describes the system model and notations. Section 4.3 discusses a centralized feasibility-optimal algorithm for real-time wireless networks. Section 4.4 presents the proposed decentralized priority-based protocol. In Section 4.5, we apply delivery debt to the proposed protocol and show that it achieves feasibility optimality. Section 4.6 provides simulation results, and Section 4.7 summarizes this chapter.

4.2 System Model and Problem Formulation

We study the problem of optimizing timely-throughputs for real-time wireless networks in a decentralized manner. The network model is described as follows.

4.2.1 Network Topology and Transmission Model

We consider a wireless ad hoc network formed by the wireless sensors, actuators, and controllers of a networked control system. To ensure connectivity for all the wireless devices, a networked control system might require multiple APs, each of which serves a subset of these client devices. These wireless entities form a set

of N directed links denoted by $\mathcal{N} = \{1, \cdots, N\}$, each of which can serve as a downlink or an uplink between an AP and a client, or a direct communication link between two clients intended for machine-to-machine interactions. For industrial control applications, we may assume that all the nodes in the network are using the same wireless protocol, and hence there is no coexistence issue. Figure 4.1 shows an example of the wireless network described above.

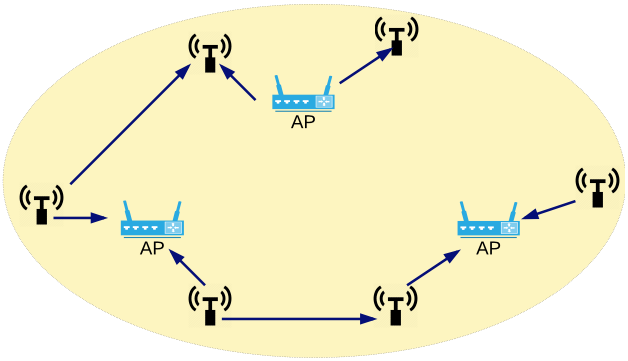


Figure 4.1: A network of multiple collocated APs serving multiple clients.

We consider the scenario where all the links share the same frequency band and interfere with each other, i.e. the conflict graph is complete. For industrial control applications, since all the wireless sensors and actuators are located in the same manufacturing area, each link is susceptible to continual interference from the other links. Since all links interfere with each other, if multiple links transmit simultaneously, a transmission collision occurs and all transmissions fail. In practice, to avoid collision due to the hidden terminal problem, the energy level of carrier sensing can be configured to be as low as possible. Moreover, transmissions can also fail due to the unreliable nature of wireless transmissions. Specifically, if link n transmits a packet and the transmission does not suffer from interference, then we assume that

the transmission is successful with probability $p_n > 0$. We use $\boldsymbol{p} = [p_n, n \in \mathcal{N}]$ to denote the success probability vector. Note that p_n can be obtained by either probing or learning from the empirical results of past transmissions. Here we simply assume that p_n is known by the transmitter of each link. If a transmission fails, the transmitter might be able to retransmit the same packet, depending on the scheduling algorithm. For simplicity, we also assume that total airtime required for transmitting a single packet (including the airtime of an ACK and the required guard time between transmissions) is the same for all the packets.

To avoid excessive collisions due to interference, *carrier sensing* is one widely-used approach (also known as *listen-before-talk*) to detect transmissions from neighboring wireless links. We assume that all the devices support carrier sensing, i.e. at each time instant each device is able to determine whether the channel is busy. However, each device is not able to overhear transmissions by other devices, since the interference range is usually much larger than the transmission range in wireless communications [90].

4.2.2 Packets with Deadlines

To continuously support real-time operations of industrial systems, each link n is associated with a data stream with a strict per-packet relative deadline equal to T time units. Note that the time unit here can be chosen arbitrarily. In many of the related works on wireless scheduling (such as [60, 66, 61]), one unit time is chosen to be the total transmission time of a packet due to the synchronized slot structure embedded in the centralized scheduling algorithms. By contrast, in this research, we do not impose the slot structure to the proposed decentralized algorithm, so that we can explicitly address the amount of time spent on channel contention. For industrial control applications, the deadline T can be chosen based on the maximum allowable

delay bound for control messages or the data sampling period. The packets that are not delivered before their deadlines are dropped. For convenience, we further divide time into *intervals*, each of which has a length of T time units and corresponds to $(kT, (k+1)T]$ for some $k \in \mathbb{N} \cup \{0\}$.

For each link, packets arrive at the beginning of each interval in a probabilistic manner. For each link n, let $A_n(k)$ be a nonnegative integer-valued random variable which denotes the number of arriving packets in the k-th interval and let $\boldsymbol{A}(k) := [A_n(k), n \in \mathcal{N}]$ be the corresponding arrival vector. For each link n and interval k, we assume that $A_n(k)$ is upper bounded by some constant $A_{\max} < \infty$. We model the arrival process $\{\boldsymbol{A}(k), k \in \mathbb{N} \cup \{0\}\}$ as a sequence of temporally independent and identically distributed (i.i.d.) random vectors with mean vector $\boldsymbol{\lambda} = [\lambda_n, n \in \mathcal{N}]$ for all interval k. Note that in each interval, the numbers of packet arrivals of different links might still be correlated. If all the packets are delivered by the end of the interval, all the links stay idle till the beginning of next interval.

In the rest of the chapter, we use $(\mathcal{N}, \boldsymbol{A}, T, \boldsymbol{p})$ to denote a wireless network described above.

4.2.3 Timely-Throughput and Feasibility Optimality

To guarantee reliable operation of the networked control system, each link n is required to maintain a minimum timely-throughput of q_n packets per interval. Equivalently, each link n requires a *minimum delivery ratio* of $\rho_n := \frac{q_n}{\lambda_n}$. Note that when there is exactly one packet arrival in each interval for each link, timely-throughput is exactly the same as delivery ratio [60]. For networked control systems with high reliability requirements, the delivery ratios are usually chosen to be close to 1.

To achieve the required timely-throughput, we focus on designing transmission

policies that determine the actions (transmit or stay idle) of each link. Let $\mathcal{H}_t$ be the history of the network up to time t. $\mathcal{H}_t$ includes all the past packet arrivals, all past actions of each link, and the outcomes of all the past transmissions. We use Π to denote the set of all the history-dependent transmission policies. For any policy $\eta \in \Pi$, it can be either randomized or deterministic, centralized or decentralized.

Now, we formally define the notion of feasibility optimality as follows. Let $S_n(k)$ be the number of delivered packets in the k-th interval, for each link n.

Definition 1. *For any $K \in \mathbb{N}$, the* timely-throughput deficiency *up to K-th interval of each link n is defined as $\left(q_n - \frac{\sum_{k=0}^{K-1} S_n(k)}{K}\right)^+$. Moreover,* total timely-throughput deficiency *up to K-th interval is defined as $\sum_{n=1}^{N}\left(q_n - \frac{\sum_{k=0}^{K-1} S_n(k)}{K}\right)^+$.*

Note that timely-throughput deficiency reflects the difference between the required timely-throughput and the empirical timely-throughput.

Definition 2. *For a network described by $(\mathcal{N}, \boldsymbol{A}, T, \boldsymbol{p})$, a timely-throughput requirement vector $\boldsymbol{q} = [q_n, n \in \mathcal{N}]$ is* fulfilled *under some policy η if total timely-throughput deficiency converges to 0 in probability as $K \to \infty$, where $(\cdot)^+ := \max\{0, \cdot\}$.*

Definition 3. *For a network described by $(\mathcal{N}, \boldsymbol{A}, T, \boldsymbol{p})$, a timely-throughput requirement vector $\boldsymbol{q} = [q_n, n \in \mathcal{N}]$ is* feasible *if there exists some policy that fulfills $\boldsymbol{q}$. Besides, $\boldsymbol{q}$ is* strictly feasible *if $q_n > 0$ for all n and there exists some $\alpha \in (0, 1)$ such that $(1 + \alpha)\boldsymbol{q}$ is also feasible.*

Definition 4. *For a network described by $(\mathcal{N}, \boldsymbol{A}, T, \boldsymbol{p})$, the* feasible region $\mathcal{Q}$ *is defined as the set of all feasible timely-throughput requirement vectors. Similarly, the* strict feasible region $\mathcal{Q}^*$ *is the set of all strict feasible timely-throughput requirement vectors.*

In this research, we focus on the strict feasible region $\mathcal{Q}^*$. Next, we introduce the definition of feasibility optimality.

Definition 5. *A policy* $\eta \in \Pi$ *is* feasibility-optimal *if it fulfills all strictly feasible* $\boldsymbol{q} \in \mathcal{Q}^*$.

Regarding the notations, we use boldface letters to denote vectors and matrices. We use $|\cdot|_\infty$ to denote the L^∞-norm of a vector or a matrix. Besides, we use $\mathbb{R}_{\geq 0}$ to denote the set of all nonnegative real numbers and use $\mathbb{P}\{\cdot\}$ to denote the probability of an event.

4.3 Centralized Feasibility-Optimal Algorithm

In this section, we study a centralized feasibility-optimal algorithm, namely the *extended largest-debt-first* (ELDF) scheduling. To begin with, we first introduce the notions of debt and debt influence functions.

4.3.1 Delivery Debt and Debt Influence Functions

To study packet deliveries with deadlines, we consider a virtual queue length, namely *delivery debt*, to capture the amount of on-time packet deliveries of each link. Recall that $S_n(k)$ denotes the number of delivered packets in the k-th interval, for each link n. Since the packets that miss their deadlines are dropped, we have $S_n(k) \leq A_n(k)$, for all n and k. Let $d_n(k)$ be the delivery debt of link n at the beginning of interval k. Then, $d_n(k)$ evolves as follows [61]:

$$d_n(k+1) = d_n(k) - S_n(k) + q_n, \tag{4.1}$$

with $d_n(0) = 0$, for all n. Equivalently, we have $d_n(k) = kq_n - \sum_{j=0}^{k-1} S_n(j)$. Based on (4.1), we know $d_n(k)$ reflects the difference between actual timely-throughput and the required timely-throughput.

To design transmission policies based on delivery debt, we consider a class of functions called *debt influence functions*.

Definition 6. *A functions $f : \mathbb{R}_{\geq 0} \to \mathbb{R}_{\geq 0}$ is said to be a* debt influence function *if*

1. *f is nondecreasing, continuous, and satisfies that $\lim_{x\to\infty} f(x) = \infty$.*

2. *Given any finite $c \in \mathbb{R}$, f satisfies that $\lim_{x\to\infty} \frac{f(x+c)}{f(x)} = 1$. Equivalently, given any $\epsilon > 0$, there exists a constant $B > 0$ such that $1 - \epsilon \leq \frac{f(x+c)}{f(x)} \leq 1 + \epsilon$, $\forall x \geq B$.*

Moreover, we use $\mathcal{F}$ to denote the set of all debt influence functions.

For example, $f(x) = x^m$ with $m \geq 0$ and $f(x) = \log_a x$ with $a > 1$ are valid debt influence functions. On the other hand, $f(x) = a^x$ with $a > 1$ is not a debt influence function.

4.3.2 A Sufficient Condition of Feasibility Optimality

We first introduce a sufficient condition which helps us design feasibility-optimal transmission policy. Note that the network can be viewed as a controlled Markov chain with inequality constraints. We then have useful results on feasibility-optimal randomized policies in the following lemma.

Lemma 10. *For any strictly feasible vector $\boldsymbol{q} \in \mathcal{Q}^*$, there exists a stationary randomized policy that fulfills $\boldsymbol{q}$ based on a probability distribution that only depends on the number of packets to be delivered and the number of time slots remaining in the current interval.*

Proof. This is a direct result of [91]. □

We first introduce a sufficient condition of feasibility optimality in the following lemma.

Lemma 11. *For a transmission policy $\eta \in \Pi$, for any debt influence function $f \in \mathcal{F}$, if given any $\delta \in (0,1)$ there exists a constant $B_0 > 0$ such that in every interval we have*

$$\mathbb{E}^{\eta}\left[\sum_{n=1}^{N} f(d_n^+(k))S_n(k) \,\middle|\, \boldsymbol{d}(k)\right] \tag{4.2}$$

$$\geq (1-\delta)\max_{\eta' \in \Pi} \mathbb{E}^{\eta'}\left[\sum_{n=1}^{N} f(d_n^+(k))S_n(k) \,\middle|\, \boldsymbol{d}(k)\right], \tag{4.3}$$

whenever $|\boldsymbol{d}(k)|_{\infty} > B_0$, then for any strictly feasible vector $\mathbf{q} \in \mathcal{Q}^*$, the Markov chain induced by $\{\mathbf{d}(k)\}$ is positive recurrent under policy η, and hence η is feasibility-optimal.

Proof. This can be proved by using Lyapunov drift analysis over one interval. Due to space limitations, the complete proof is provided in Appendix C.1. □

Remark 16. *Note that by choosing $f(x) = x$ in Lemma 11, we can recover Theorem 2 in [61], which considers only linear debt influence function. Besides, Lemma 11 serves a similar purpose as Claim 1 in [92]. Different from [92], we consider packets with deadlines in this research.*

4.3.3 A Feasibility-Optimal Centralized Scheduling Algorithm

Motivated by the *Largest-Debt-First* (LDF) scheduling policy proposed in [60, 61], we consider the following extended version of LDF (ELDF) as shown in Algorithm 2.

Algorithm 2 Extended Largest-Debt-First Scheduling Policy

1: At the beginning of the k-th interval, sort all of the N clients such that

$$f(d_{m_1}^+(k))p_{m_1} \geq f(d_{m_2}^+(k))p_{m_2} \geq \cdots \geq f(d_{m_N}^+(k))p_{m_N}. \tag{4.4}$$

In this way, $[m_n, n \in \mathcal{N}]$ determines the transmission priority of each link.

2: Transmit packets based on the priority ordering of (4.4) till the end of the k-th interval.

Remark 17. *Note that ELDF is a priority-based policy, which updates its priority ordering at the beginning of each interval. By choosing $f(x) = x$, the ELDF policy becomes equivalent to the LDF policy, as both of them assign priorities according to $\boldsymbol{d}(k)$.*

Next, we show that the ELDF policy is feasibility-optimal.

Lemma 12. *Under unreliable channels described in Section 4.2.1, the ELDF policy using delivery debt maximizes $\mathbb{E}\Big[\sum_{n=1}^{N} f(d_n^+(k))S_n(k) \;\Big|\; \boldsymbol{d}(k)\Big]$ among all the policies in Π.*

Proof. This is a direct result of Theorem 3 in [61]. □

Based on Lemma 12, we know that ELDF always satisfies the sufficient condition provided in Lemma 11.

Proposition 1. *Under unreliable channel model and delivery debt $\{\boldsymbol{d}(k)\}$, the ELDF policy is feasibility-optimal.*

Proof. This can be proved by Lemma 11 and Lemma 12. □

4.4 Decentralized Priority-Based Protocol

While Section 4.3 has introduced the feasibility-optimal centralized algorithm, such an algorithm may be infeasible to implement in a fully decentralized fashion. Specifically, Algorithm 2 requires the complete knowledge of the number of packets

generated by each link, as well as the delivery debt of each link. On the other hand, we also note that Algorithm 2 has the nice structure of being a priority-based policy, that is, it assigns a priority to each link at the beginning of each interval. In this section, we propose a generic decentralized priority-based algorithm and describe the features of the proposed algorithm.

4.4.1 Transmission Priorities and Permutations

To deal with transmission priorities, we introduce some useful definitions regarding permutations as follows.

Definition 7. *For a set of integers $\mathcal{N} = \{1, \cdots, N\}$, a* permutation $\boldsymbol{\sigma}$ *of $\mathcal{N}$ is a bijective map from $\mathcal{N}$ to $\mathcal{N}$.*

With a slight abuse of notation, we represent a permutation $\boldsymbol{\sigma}$ in vector form as $\boldsymbol{\sigma} = [\sigma_1, \cdots, \sigma_N]$. Next, we consider the transition from one permutation to another.

Definition 8. *A* transposition *of a permutation $\boldsymbol{\sigma}$ is defined as an exchange of two entries in $\boldsymbol{\sigma}$. Moreover, an* adjacent transposition *(i, j) of $\boldsymbol{\sigma}$ is defined as an exchange of two entries σ_i and σ_j with $|\sigma_i - \sigma_j| = 1$, for some $i, j \in \mathcal{N}$.*

Definition 9. *For any two permutations $\boldsymbol{\sigma}, \boldsymbol{\sigma}' \in \mathfrak{S}_N$, the* symmetric difference *is defined as $\boldsymbol{\sigma} \triangle \boldsymbol{\sigma}' := \{n : \sigma_n \neq \sigma'_n\}$.*

Example 1. *Let $N = 4$ and consider two permutations: $\boldsymbol{\sigma} = [2, 1, 4, 3]$, $\boldsymbol{\sigma}' = [2, 4, 1, 3]$. By definition, we have $\boldsymbol{\sigma} \triangle \boldsymbol{\sigma}' = \{2, 3\}$, and $\boldsymbol{\sigma}'$ can be obtained by applying an adjacent transposition $(2, 3)$ to $\boldsymbol{\sigma}$.*

In the rest of the chapter, we use $\mathfrak{S}_N$ to denote the set of all permutations of $\{1, 2, ..., N\}$. In each interval k, let $\boldsymbol{\sigma}(k) = [\sigma_1(k), ..., \sigma_N(k)]$ be the transmission priority vector, where each $\sigma_n(k)$ denotes the priority index of link n, and $\boldsymbol{\sigma}(k)$ can

be any permutation in $\mathfrak{S}_N$. The link with the highest priority should have priority index equal to 1.

4.4.2 A Generic Decentralized Priority-Based Protocol

In this section, we introduce a generic decentralized priority-based (DP) protocol as shown in Algorithm 3. In the DP protocol, the transmissions are designed to be collision-free by making different links choose different backoff timers in each interval. Specifically, in each interval, each link has a unique priority index within the range $\{1, \cdots, N\}$ and chooses its backoff timer based on this priority, as shown in Step 4. Note that this design is completely decentralized in the sense that each link only needs to know its own priority index.

To dynamically adjust priorities, in each interval, a pair of links with priority difference equal to 1 (denoted by $C(k)$ and $C(k)+1$ in Algorithm 3) is chosen uniformly at random as candidates for swapping priorities. These two links swap priorities only when the link of priority $C(k)$ tend to move down and the link of priority $C(k)+1$ tend to move up. Without using any control message, the above event can be detected by both links via carrier sensing plus proper choices of backoff timers. As shown in Step 3 and 4, each of the two swapping candidates determines its backoff timer by using an individual coin toss with parameter μ_n. As shown in (4.7), if the link of priority $C(k)$ tends to move down, it increases its priority index by 1 only if the channel is sensed busy when backoff number decreases to 1. The event that the channel is busy when backoff number decreases to 1 indicates that the link of priority $C(k)+1$ also tends to move up. Similarly, as described in (4.8), if the link of priority $C(k)+1$ tends to move up, it detects that the link of priority $C(k)$ also tends to move down only if the channel is sensed idle when backoff number decreases to 1. Any change in priority index will be enforced in the next interval. By

using the above mechanism, the two candidate links achieve implicit coordination completely via carrier sensing. For any non-candidate link, it simply remains at the same priority for one interval.

Algorithm 3 Decentralized Priority (DP) Protocol With Randomized Reordering (Link n in the k-th interval)

1: At the beginning of the k-th interval, select an integer-valued uniform random variable $C(k)$ with value between 1 and $N-1$ by using a random seed shared by all the devices (for example, system time).

2: If $n \in \{C(k), C(k)+1\}$ and link n has no packet arrival in the current interval, then link n generates an empty packet (for the purpose of priority claiming) and puts it in its buffer.

3: Generate a random variable $\xi_n(k)$ locally as

$$\xi_n(k) = \begin{cases} 1 & \text{, with probability } \mu_n \\ -1 & \text{, with probability } 1-\mu_n \end{cases} \tag{4.5}$$

where $\mu_n \in (0,1)$ is a parameter chosen by link n.

4: Given the priority index $\sigma_n(k-1)$, determine the backoff timer $\beta_n(k)$ for the current interval as

$$\beta_n(k) = \begin{cases} \sigma_n(k-1) - 1 & \text{, if } \sigma_n(k-1) < C(k) \\ \sigma_n(k-1) + 1 & \text{, if } \sigma_n(k-1) > C(k)+1 \\ \sigma_n(k-1) - \xi_n(k) & \text{, if } \sigma_n(k-1) = C(k) \\ & \quad \text{or } \sigma_n(k-1) = C(k)+1 \end{cases} \tag{4.6}$$

5: At any time instant, if link n does not hear any transmission from any other link,

link n continues on the backoff procedure. If $n = C(k)$, then link n updates its priority index by

$$\sigma_n(k) = \begin{cases} \sigma_n(k-1) & , \text{if } \xi_n(k) = 1 \\ \sigma_n(k-1)+1 & , \text{if } \xi_n(k) = -1 \text{ and channel is} \\ & \text{busy when backoff timer} \\ & \text{decreases to 1} \\ \sigma_n(k-1) & , \text{otherwise} \end{cases} \tag{4.7}$$

If $n = C(k)+1$, then link n updates its priority index by

$$\sigma_n(k) = \begin{cases} \sigma_n(k-1)-1 & , \text{if } \xi_n(k) = 1 \text{ and channel is} \\ & \text{idle when backoff timer} \\ & \text{decreases to 1} \\ \sigma_n(k-1) & , \text{otherwise} \end{cases} \tag{4.8}$$

If n is neither $C(k)$ nor $C(k)+1$, then link n simply updates its priority index by $\sigma_n(k) = \sigma_n(k-1)$.

6: When the backoff timer decreases to zero, link n starts transmitting packets till the end of the k-th interval or until there is no packet left in its buffer.

7: At the end of the current interval, flush all the packets in the buffer and updates network states accordingly.

Remark 18. *Note that the DP protocol does not have the synchronized slot structure embedded in many centralized scheduling algorithms (such as [60, 66, 61]). In fact, there might be a short period of idle time due to backoff between two packet*

transmissions. To make sure that every link obtains the same $C(k)$ in Step 1, the DP protocol only requires a common random seed (e.g. through initial coarse time synchronization).

Remark 19. *In Step 6 of the DP protocol, if the remaining time of the current interval is less than the transmission time of a packet, link n simply stays idle till the end of the interval. This "gap" is less than one packet transmission time and can be minimized by choosing a proper combination of packet deadline and packet transmission time.*

Remark 20. *In Step 3 of the DP protocol, each link needs to choose the parameter μ_n. In Section 4.5, we will discuss how to choose μ_n to achieve feasibility optimality.*

Remark 21. *Note that the DP protocol can be further generalized to the cases of multiple swapping pairs. Specifically, in Step 1 of Algorithm 3, multiple non-consecutive integers are selected among $\{1, \cdots, N-1\}$, and in Step 4 the backoff timers are determined in a similar manner. Please refer to [93] for more detailed discussions.*

To make the idea of DP protocol more clear, we consider the following toy example:

Example 2. *Consider a network of 4 links $\{1, 2, 3, 4\}$, each of which has channel reliability $p_n = 1$ and exactly 1 packet arrival at the beginning of each interval. Suppose $\boldsymbol{\sigma}(1) = [1, 2, 3, 4]$ and $\boldsymbol{\sigma}(2) = [1, 3, 2, 4]$. As shown in Figure 4.2, Link 2 and 3 exchange priorities if link 2 and link 3 set backoff timer $\beta_2(k) = 3$ and $\beta_3(k) = 2$ in Step 4 of Algorithm 3, respectively.*

Note that Step 1 to 6 in Algorithm 3 are completely agnostic to packet deadlines. Therefore, Algorithm 3 actually provides a generic approach to implementing priority-based algorithms in a fully-decentralized fashion.

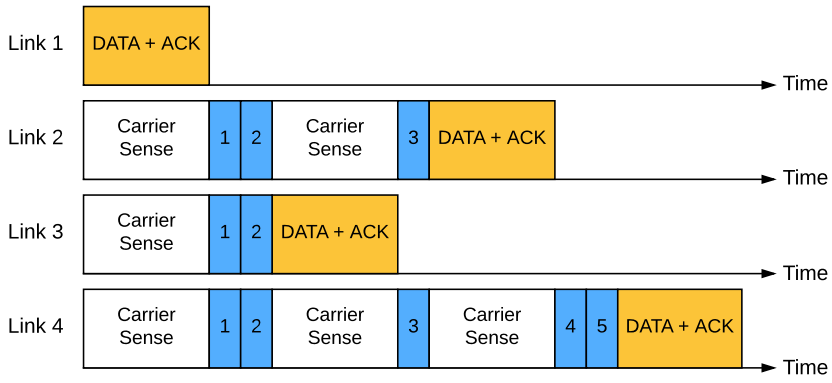


Figure 4.2: An example of priority exchange using backoff.

4.4.3 Features of the Decentralized Protocol

Before providing theoretical analysis of the decentralized protocol, we first highlight the important features of the protocol.

- **Fully-decentralized**: Under the DP protocol, each link only needs to know its own priority. To update the priority index, each link only needs to determine whether the channel is busy by using carrier sensing. This completely removes the messaging overhead of maintaining network states at an AP required by a centralized scheduling algorithm.

- **Quantifiably small overhead incurred by maintaining priorities**. One advantage of the proposed algorithm is that the overhead can be clearly quantified. The overhead comes from two main parts: (i) *backoff timer*: According to Step 4 of Algorithm 3, we know that the backoff timer of each link is at most $N+1$. For example, in the widely-used 802.11a/g/n/ac standard, one backoff slot is chosen to be 9 μs, which is usually much smaller than the packet deadline (several milliseconds). Moreover, even shorter backoff slot time is also possible by optimizing carrier sensing functionality and Rx to Tx turnaround time [94]. (ii)

empty packet for claiming priority: In each interval, there are at most two empty packets, each of which requires much smaller transmission time than that of a typical data packet. For example, in 802.11a protocol, the transmission time of a packet with no payload plus the required interframe spacing is about 70 μs. By contrast, even with the highest data rate of 54 Mbps, the time to send a typical 1500 B data packet plus an ACK is roughly 330 μs [95]. Therefore, the overhead incurred by an empty packet is indeed small.

- **No capacity loss due to collision**. The DP protocol enforces transmission priority by letting each link keep a unique backoff number in each interval. In this way, there is no channel contention issue common to many other CSMA-based algorithms that utilize a random backoff mechanism. Therefore, the proposed protocol completely obviates the capacity loss due to collided transmissions. This is particularly critical for industrial networked control systems, where excessive collisions could happen continually due to massive connectivity.

- **Requires only coarse carrier sensing and initial synchronization**. The DP protocol utilizes the discrete backoff mechanism and therefore requires only coarse carrier sensing (or equivalently, carrier sensing can be non-instantaneous). In order to synchronize packet arrivals as other CSMA-based algorithms (such as [76, 80, 81]), the proposed algorithm assumes initial time synchronization, which can be easily achieved by having APs broadcast messages or simply using GPS time information if available. Therefore, the proposed protocol can be easily implemented in the wireless devices without any additional hardware.

- **No control packets or control slots required**. Different from the Q-CSMA-like algorithms [76, 81], there is no control packet required in the proposed DP

protocol. This design greatly reduces the messaging overhead as well as implementation efforts.

- **Robust to packet losses**. Note that under the DP protocol, transmission priorities are maintained by backoff timers and transmission attempts (regardless of the outcome), not control packets. Therefore, the DP protocol is robust to packet losses due to unreliable channels.

4.4.4 Analysis of Stationary Distribution

In this section, we study the behavior of the proposed decentralized protocol in steady state. Note that the stochastic process $\{\boldsymbol{\sigma}(k), k \in \mathbb{N}_0\}$ can be modeled as a discrete-time Markov chain with a finite state space $\mathfrak{S}_N$ and a stationary $N! \times N!$ transition probability matrix $\boldsymbol{X} = [X_{\boldsymbol{\sigma},\widehat{\boldsymbol{\sigma}}}, \boldsymbol{\sigma}, \widehat{\boldsymbol{\sigma}} \in \mathfrak{S}_N]$. Let $\boldsymbol{\sigma}, \widehat{\boldsymbol{\sigma}}$ be two permutations in $\mathfrak{S}_N$. If $\boldsymbol{\sigma} \triangle \widehat{\boldsymbol{\sigma}} = \{i, j\}$ for some $i, j \in \mathcal{N}$ and (i, j) forms an adjacent transposition, we have

$$\boldsymbol{X}_{\boldsymbol{\sigma},\widehat{\boldsymbol{\sigma}}} = \frac{(1-\mu_i)\mu_j}{N-1} \cdot \mathbb{P}\{R_i + R_j \geq 1\}, \tag{4.9}$$

where R_i and R_j denote the number of transmissions (including empty packets) made by link i and j in the current interval, respectively. Otherwise, $\boldsymbol{X}_{\boldsymbol{\sigma},\widehat{\boldsymbol{\sigma}}} = 0$. In (4.9), the term $1/(N-1)$ comes from randomization (Step 1 of Algorithm 3), the term $(1-\mu_i)\mu_j$ represents the probability of the event that the candidate links tend to exchange priority indices (Step 3 and 4 of Algorithm 3), and $\mathbb{P}\{R_i + R_j \geq 1\}$ represents the probability of the event that at least one of the candidate links needs to transmit before confirming the exchange of priority indices (Step 5 of Algorithm 3). To ensure that $\boldsymbol{X}_{\boldsymbol{\sigma},\widehat{\boldsymbol{\sigma}}}$ in (4.9) is nonzero, we impose a mild technical condition as follows:

- (C1) With a non-zero probability, the total number of packet arrivals in one interval is less than the number of available transmission attempts in the interval.

Next, we present an important property of the Markov chain $\{\boldsymbol{\sigma}(k)\}$ as below.

Lemma 13. *Under Algorithm 3 with condition (C1), the Markov chain* $\{\boldsymbol{\sigma}(k)\}$ *is irreducible and aperiodic.*

Proof. Due to space limitations, the complete proof is provided in Appendix C.2. □

Based on Lemma 13, we are able to derive the stationary distribution of the Markov chain $\{\boldsymbol{\sigma}(k)\}$ under Algorithm 3.

Proposition 2. *Under Algorithm 3 with condition (C1), the stationary Markov chain* $\{\boldsymbol{\sigma}(k)\}$ *is time-reversible and has the unique stationary distribution* π^* *as*

$$\pi^*(\boldsymbol{\sigma}) = \frac{\prod_{n=1}^{N} \left(\frac{\mu_n}{1-\mu_n}\right)^{g(\sigma_n)}}{Z}, \ \ \forall \boldsymbol{\sigma} \in \mathfrak{S}_N \tag{4.10}$$

where

$$Z := \sum_{\boldsymbol{\sigma} \in \mathfrak{S}_N} \prod_{n=1}^{N} \left(\frac{\mu_n}{1-\mu_n}\right)^{g(\sigma_n)} \tag{4.11}$$

$$g(j) := \begin{cases} N-j, & if\, 1 \le j \le N \\ 0, & otherwise \end{cases} \tag{4.12}$$

Proof. To show that $\{\boldsymbol{\sigma}(k)\}$ is time-reversible, we simply verify that (4.10)-(4.12) satisfy the detailed balance equations [96]. For any two transmission priority vectors

$\boldsymbol{\sigma} = (\sigma_1, \cdots, \sigma_N)$ and $\widetilde{\boldsymbol{\sigma}} = (\widetilde{\sigma}_1, \cdots, \widetilde{\sigma}_N)$, if the symmetric difference $\boldsymbol{\sigma} \triangle \widetilde{\boldsymbol{\sigma}} = \{i, j\}$ and $\sigma_i = \widetilde{\sigma}_j = m$ and $\sigma_j = \widetilde{\sigma}_i = m + 1$, then we have

$$\frac{\pi^*(\boldsymbol{\sigma})}{\pi^*(\widetilde{\boldsymbol{\sigma}})} = \frac{\prod_{n=1}^{N} \left(\frac{\mu_n}{1-\mu_n}\right)^{g(\sigma_n)}}{\prod_{n=1}^{N} \left(\frac{\mu_n}{1-\mu_n}\right)^{g(\widetilde{\sigma}_n)}} = \frac{\frac{\mu_i}{1-\mu_i}}{\frac{\mu_j}{1-\mu_j}}. \tag{4.13}$$

Since $\boldsymbol{\sigma}$ and $\widetilde{\boldsymbol{\sigma}}$ share the same transmission ordering for priority 1 through $m-1$, then $\mathbb{P}\{R_i + R_j \geq 1\}$ is the same under $\boldsymbol{\sigma}$ and $\widetilde{\boldsymbol{\sigma}}$. By (4.9) and (4.13), we know that $\pi^*(\boldsymbol{\sigma}) \cdot \boldsymbol{X}_{\boldsymbol{\sigma},\widetilde{\boldsymbol{\sigma}}} = \pi^*(\widetilde{\boldsymbol{\sigma}}) \cdot \boldsymbol{X}_{\widetilde{\boldsymbol{\sigma}},\boldsymbol{\sigma}}$. By Lemma 13, we know $\{\boldsymbol{\sigma}(k)\}$ is irreducible and aperiodic, and therefore the stationary distribution is unique. □

4.5 Decentralized Priority Algorithm For Feasibility Optimality

In this section, we elaborate on how to achieve feasibility optimality by choosing proper parameters for the randomized reordering in Step 3 of the DP protocol.

4.5.1 A Decentralized Priority Algorithm Using Delivery Debt

In the ELDF policy, transmission priorities are determined by the product of the value of debt influence function and the channel reliability. Based on this observation, we shall choose μ_n as a function of delivery debt $d_n(k)$ such that μ_n increases with $d_n(k)$. Motivated by the Glauber dynamics [71], we choose μ_n for Step 3 in Algorithm 3 as

$$\mu_n(k) = \frac{\exp(f(d_n^+(k))p_n)}{R + \exp(f(d_n^+(k))p_n)}, \tag{4.14}$$

where f is the debt influence function and R is a predetermined positive constant. Note that similar forms to (4.14) have also been adopted in [73, 76, 80]. Meanwhile, in order to apply the results of Lemma 13 and Proposition 2, the Markov chain $\{\boldsymbol{\sigma}(k)\}$ is required to be stationary. Here we consider the concept of *two-time-scale separation* for the Markov chain $\{\boldsymbol{\sigma}(k)\}$. Specifically, we choose μ_n to be

a slowly-changing function of $d_n(k)$ such that the evolution of μ_n is much slower than the evolution of $\{\boldsymbol{\sigma}(k)\}$. For example, [71] conjectures that $f(x) = \log\log x$ achieves sufficiently slow evolution of the underlying Markov chain. This result is later relaxed to $f(x) \approx \log(x)$ by [75] for better convergence time. Besides, [76] also shows via simulation that $f(x) = \log x$ achieves the best delay performance. In this way, we are able to approximate the Markov chain $\{\boldsymbol{\sigma}(k)\}$ by its "quasi-stationary" characteristics. This argument has been used extensively in the existing literature for both continuous-time and discrete time Markov chains, such as [97, 71, 73, 76, 98]. Therefore, we rely on this assumption to show feasibility optimality of the proposed algorithm for brevity.

In the rest of the chapter, we will refer to Algorithm 3 with $\mu_n(k)$ in (4.14) as the *debt-based decentralized priority algorithm* (DB-DP). Next, we summarize the important properties of the Markov chain $\{\boldsymbol{d}(k)\}$ under the DB-DP algorithm.

Proposition 3. *Under the DB-DP algorithm, the Markov chain $\{\boldsymbol{\sigma}(k)\}$ is irreducible and aperiodic. Moreover, the Markov chain $\{\boldsymbol{\sigma}(k)\}$ is reversible and has the unique stationary distribution π^* as*

$$\pi^*(\boldsymbol{\sigma}, k) = \frac{\exp\left(\sum_{n=1}^{N} g(\sigma_n) f(d_n^+(k)) p_n\right)}{Z(\boldsymbol{d}(k))}, \quad \forall \boldsymbol{\sigma} \in \mathfrak{S}_N \tag{4.15}$$

where

$$Z(\boldsymbol{d}(k)) := \sum_{\boldsymbol{\sigma}\in\mathfrak{S}_N} \exp\left(\sum_{n=1}^{N} g(\sigma_n) f(d_n^+(k)) p_n\right) \tag{4.16}$$

$$g(j) := \begin{cases} N-j, & \text{if } 1 \le j \le N \\ 0, & \text{otherwise} \end{cases} \tag{4.17}$$

Proof. Under two-time-scale separation, (4.15)-(4.17) can be shown by directly substituting (4.14) for μ_n in (4.10)-(4.12). □

4.5.2 Proof of Feasibility Optimality

In this section, we show that the DB-DP algorithm indeed achieves feasibility optimality. Recall that in Section 4.4.3, we explicitly quantify the overhead incurred by maintaining priorities. For ease of exposition, in this section, we assume that the time of each backoff slot and the time to transmit an empty packet are zero. We introduce the following definition:

Definition 10. *The proposed decentralized priority-based protocol is said to be* idealized *if the time of each backoff slot and the time to transmit an empty packet are both zero.*

Accordingly, we let the packet deadline to be as long as T packet transmissions. In Section 4.6, we will provide more detailed discussion on how to determine packet deadlines, packet transmission time, and backoff slots, etc.

Proposition 4. *Under the idealized DB-DP algorithm (denoted by η), for any*

$\delta \in (0,1)$, *there exists a* $B > 0$ *such that in every interval we have*

$$\mathbb{E}^{\eta}\left[\sum_{n=1}^{N} f(d_n^+(k))S_n(k) \,\middle|\, \boldsymbol{d}(k)\right] \tag{4.18}$$

$$\geq (1-\delta)\max_{\eta' \in \Pi} \mathbb{E}^{\eta'}\left[\sum_{n=1}^{N} f(d_n^+(k))S_n(k) \,\middle|\, \boldsymbol{d}(k)\right], \tag{4.19}$$

whenever $|\boldsymbol{d}(k)|_{\infty} > B$.

Proof. We show (4.18)-(4.19) by using the stationary distribution obtained by Proposition 2. Different from the ELDF policy, the DB-DP algorithm does not maintain the "optimal" transmission ordering of (4.4) with probability 1. In spite of this, we can still show that the transmission priority ordering under the DB-DP algorithm is close to "optimal" with high probability when delivery debts are sufficiently large. Due to space limitations, we include the complete proof in Appendix C.3. □

Remark 22. *The main challenge of proving Proposition 4 is two-fold: (i) We need to calculate (4.18)-(4.19) based on transmission priority ordering instead of the exact schedules. It is not immediately clear how to compare different priority orderings in terms of (4.18). This is an essential difference between our proof and conventional drift analysis for MaxWeight-type scheduling policies. (ii) The transmission priority ordering is stochastic. Therefore, we need to consider all possible priority orderings.*

Theorem 20. *The idealized DB-DP algorithm is feasibility-optimal.*

Proof. This can be proved by Lemma 11 and Proposition 4. □

4.6 Simulation Results

In this section, we evaluate the proposed algorithm via extensive NS-3 simulations. NS-3 provides a discrete-event environment, which allows us to implement all

the features of the DB-DP algorithm. The source code for the simulations can be found at [99]. We are particularly interested in two applications: (i) real-time video delivery and (ii) ultra-low-latency control information delivery. While these two applications both require deadline guarantees, they greatly differ in traffic pattern as well as the required level of wireless service.

Throughout the simulations, we consider IEEE 802.11a as the underlying MAC protocol with physical data rate equal to 54 Mbps. Under 802.11a, a backoff slot is set to be 9 μs to account for non-instantaneous carrier sensing. We compare the proposed DB-DP algorithm with the LDF policy (a special case of ELDF with $f(x) = x$) and the FCSMA algorithm proposed by [80]. For the DB-DP algorithm, we choose the debt influence function as $f(x) = \log(\max\{1, 100(x+1)\})$ and $R = 10$. For the FCSMA algorithm, we consider its discretized version with the same parameters as suggested in [80]. We evaluate the performance of the three algorithms based on the *total timely-throughput deficiency* defined in Definition 1. By Definition 2, we know that under a given timely-throughput requirement vector $\mathbf{q}$, $\mathbf{q}$ is fulfilled if and only if the total timely-throughput deficiency converges to zero as simulation time goes to infinity. Note that in the following simulations, we might observe a small non-zero total timely-throughput deficiency even for feasible $\mathbf{q}$ due to the finite simulation time.

4.6.1 Real-Time Video Delivery

In this section, we provide simulation results for real-time video delivery, which is required by many industrial networked control systems for machine vision and process surveillance [100]. We assume the payload size of each data packet is 1500 B and the packet deadline is 20 ms. The total airtime required by sending a data packet plus an ACK and the interframe spacing is about 330 μs. Under LDF, there

are up to 60 transmissions in each interval. On the other hand, under the proposed DB-DP algorithm, there might be 1 or 2 fewer transmissions in each interval due to the time spent on backoff slots and empty packets for claiming priority. To capture the bursty traffic pattern of video streams, we assume that the number of packet arrivals at each link n in each interval is uniformly distributed within $\{1, 2, 3, 4, 5, 6\}$ with probability α_n and is 0 with probability $1 - \alpha_n$. Therefore, the arrival rate $\lambda_n = 3.5\alpha_n$, for each link n. The simulation time is 5000 intervals, or equivalently 100 seconds.

First, we consider a fully-symmetric network of 20 links, i.e. all the links have the same channel reliability $p_n = p*$, arrival rate λ_n, and delivery ratio ρ_n. Accordingly, we let $\alpha_n = \alpha^*$, for all n. We assume that $p* = 0.7$ and the delivery ratio $\rho_n = 0.9$. Figure 4.3 shows the total timely-throughput deficiency under three algorithms. Note that the proposed DB-DP algorithm achieves almost the same performance as LDF, which is known to be a feasibility-optimal centralized algorithm. On the other hand, FCSMA supports only about 70% of the maximum admissible α^* (about 0.62 based on the results of LDF in Figure 4.3). This is mainly due to the capacity loss resulting from significant backoff overhead as well as collided transmissions as discussed in Section 4.1. Next, we study the achievable delivery ratio under a fixed arrival rate. We assume $\alpha^* = 0.55$, which is slightly smaller than the maximum admissible α^* suggested by Figure 4.3. Figure 4.4 shows the total timely-throughput deficiency under $\alpha^* = 0.55$ and various delivery ratios. Similar to Figure 4.3, we observe that DB-DP and LDF achieve almost the same level of delivery ratio while FCSMA suffers from significant loss of timely-throughput.

To show the convergence time under DB-DP and LDF, we consider the timely-throughput of the link with the lowest priority at time 0 under $\alpha^* = 0.55$ and 93% delivery ratio, as shown in Figure 4.5. As expected, LDF indeed achieves a relatively

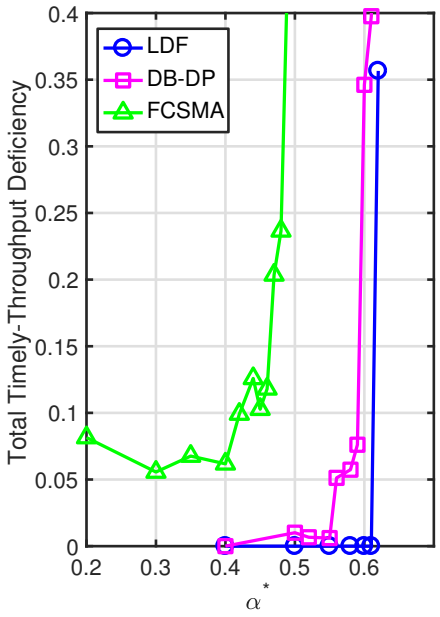


Figure 4.3: Total timely-throughput deficiency of the symmetric network under 90% delivery ratio.

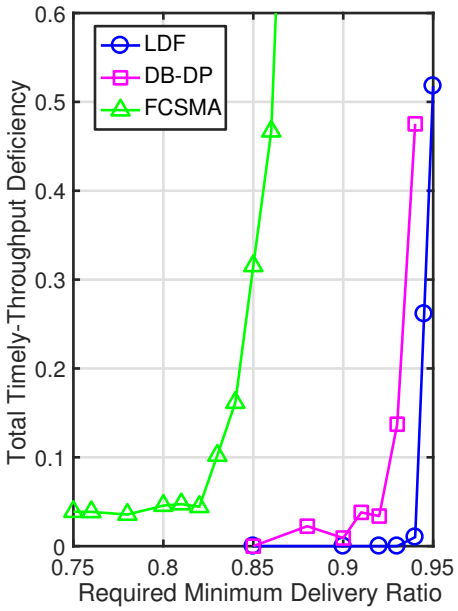


Figure 4.4: Total timely-throughput deficiency of the symmetric network under a fixed arrival rate with $\alpha^* = 0.55$.

small convergence time due to the nature of centralized control. We also observe that DB-DP achieves a convergence time (e.g. within 1% neighborhood of the timely-throughput requirement) comparable to that of LDF policy. As mentioned in Section 4.1, the DB-DP algorithm naturally alleviates the starvation problem of conventional CSMA algorithms due to the design of priority structure. To make this more clear, Figure 4.6 shows the average timely-throughput of each link under a *fixed priority ordering* and $\alpha^* = 0.6$. As expected, the average timely-throughput increases with priority (with small variations due to random arrivals) and the link with the lowest priority (priority index = 20) still receives non-zero timely-throughput. Further results on convergence time can be found in the technical report [93].

Next, we consider an asymmetric scenario by dividing the 20 links into two groups

of equal size:

- Group 1: each link has $p_n = 0.5$ and $\alpha_n = 0.5\alpha^*$.
- Group 2: each link has $p_n = 0.8$ and $\alpha_n = \alpha^*$.

The required delivery ratio is still 0.9 for each link. Figure 4.7 and Figure 4.8 show the group-wide total timely-throughput deficiency under a fixed delivery ratio 90% and a fixed $\alpha^* = 0.7$, respectively. Similar to Figure 4.3 and 4.4, DB-DP algorithm still achieves almost the same performance as the LDF policy with the existence of network asymmetry. Note that under FCSMA, group 1 suffers from much larger deficiency than group 2. This is mainly due to the discretization design of [80], where the range of delivery debt is divided into a finite number of sections and each section is mapped to one of the predetermined sizes of the contention window. Therefore, the size of contention window is the same for any delivery debt above a certain threshold. Hence, FCSMA becomes completely oblivious when delivery debt is sufficiently large and fails to respond to the changes in delivery debt.

4.6.2 Ultra-Low-Latency Control Information Delivery

In this section, we present simulation results for ultra-low-latency information delivery, which is critical to industrial networked control systems. In order for the machines to reliably perform mission-critical tasks, the machines and the controllers need to continuously exchange time-critical control messages, which require a per-packet deadline below 10 ms [101, 102]. These control packets are usually small (less than 100 B) but require much more stringent per-packet deadlines than other data packets. Here we assume the per-packet deadline T is 2 ms and the payload size of each packet is 100 B. Under the same MAC-layer settings, the total airtime to transmit one packet plus an ACK is roughly 120 μs. Regarding the arrival process,

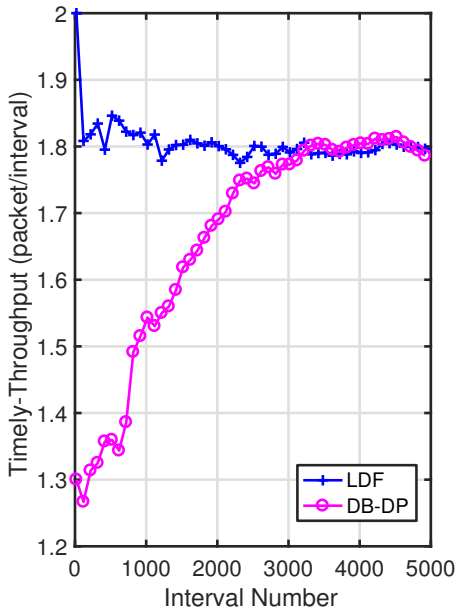


Figure 4.5: Comparison of convergence time under DB-DP and LDF policy with $\alpha^* = 0.55$ and 93% delivery ratio.

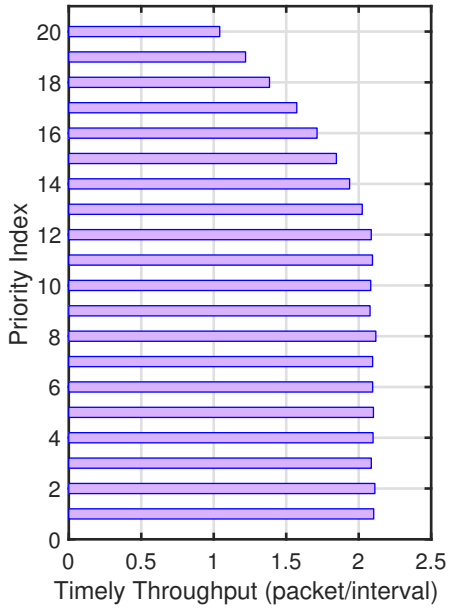


Figure 4.6: Average timely-throughput under a fixed priority ordering, $\alpha^* = 0.6$.

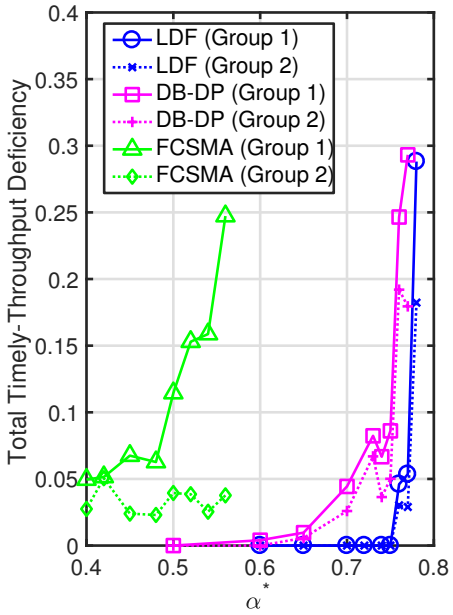


Figure 4.7: Group-wide total timely-throughput deficiency of asymmetric network under 90% delivery ratio.

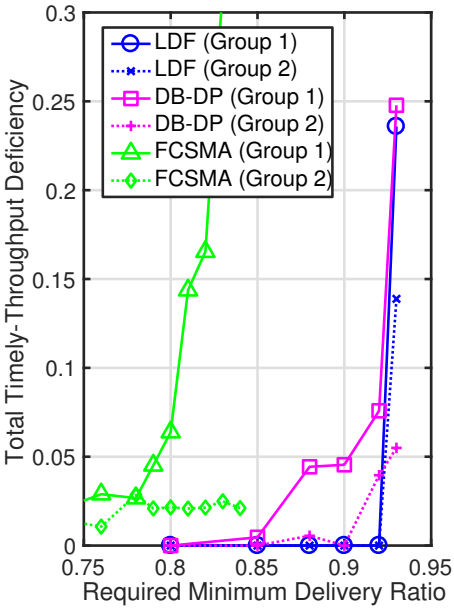


Figure 4.8: Group-wide total timely-throughput deficiency of asymmetric network under $\alpha^* = 0.7$.

we assume that the numbers of packet arrivals of each link in each interval form a sequence of i.i.d. Bernoulli random variables with mean λ^*. The total simulation time is 20000 intervals, or equivalently 40 seconds.

For ease of exposition, we consider a fully-symmetric network of 10 links, where each link has the same $p_n = 0.7$, $\lambda_n = \lambda^*$, and the delivery ratio equal to 0.99. Figure 4.9 and Figure 4.10 show the total timely-throughput deficiency under a fixed delivery ratio 99% and under a fixed $\lambda^* = 0.78$, respectively. Note that under LDF there are 16 available transmissions in each interval. On the other hand, DB-DP might have 1 or 2 fewer transmissions in one interval due to the overhead of backoff slots and empty packets. In spite of this, DB-DP still achieves a timely-throughput close to that of the LDF policy when the packet deadline is as low as 2 ms.

Based on the discussion in Section 4.6.1 and 4.6.2, we see that the DB-DP indeed achieves the same level of real-time wireless service as the LDF policy.

4.7 Summary

This chapter presents a feasibility-optimal decentralized algorithm for real-time wireless ad hoc networks over unreliable wireless channels. We first present a generic decentralized priority-based protocol that utilizes only carrier sensing and collision-free backoff mechanism. Under the proposed protocol, the overhead of maintaining transmission priority ordering is small and can be easily quantified. Next, we combine the generic decentralized protocol with delivery debt and show that it is feasibility-optimal. We evaluate the performance of the proposed decentralized algorithm via extensive NS-3 simulations and show that it performs as well as the feasibility-optimal centralized algorithm.

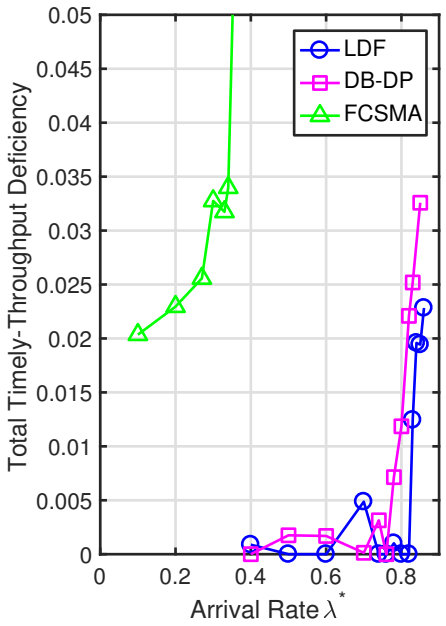


Figure 4.9: Total timely-throughput deficiency under 99% delivery ratio.

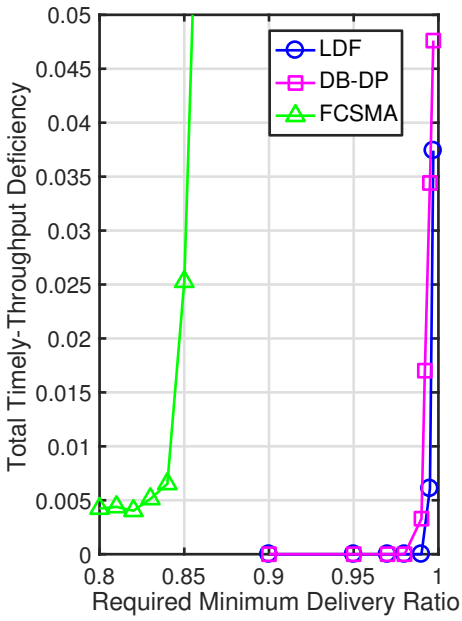


Figure 4.10: Total timely-throughput deficiency under a fixed $\lambda^* = 0.78$.

5. A LOW-LATENCY SOFTWARE-DEFINED WIRELESS TESTBED FOR PROTOTYPING WIRELESS PROTOCOLS[1]

In this chapter, we introduce a low-latency software-defined wireless testbed for prototyping wireless network algorithms. Recall that in Chapter 3.8, we provide experimental results done on the proposed testbed for QoE-optimal scheduling policies. Here we shall focus mainly on the implementation of real-time wireless network algorithms.

5.1 Background and Motivation

Strict latency requirement is currently one of the most critical challenges for next-generation wireless networks. Emerging applications, such as virtual reality (VR) [104] and industrial IoT [105], require an end-to-end latency between 1 to 10 milliseconds to provide seamless user experience and reliable real-time control. However, the existing wireless networks cannot provide such stringent latency guarantees. For example, the round-trip time of LTE is estimated to be at least 20 milliseconds, including the transmission time, scheduling overhead, and processing delay [106]. In practice, the current LTE technology can only support voice or video streaming applications with round-trip time in the range of 20-60 milliseconds [107]. For Wi-Fi networks, due to the nature of random access, the round-trip time could vary from several milliseconds to hundreds of milliseconds depending on the traffic load [108]. Therefore, compared to the current technology, the latency budget is expected to be at least one order of magnitude smaller in next-generation wireless networks.

[1]Part of this chapter is reprinted with permission from "PULS: Processor-Supported Ultra-Low Latency Scheduling" and "Demo: PULS: Processor-Supported Ultra-Low Latency Scheduling" by Simon Yau, Ping-Chun Hsieh, Rajarshi Bhattacharyya, Kartic Bhargav K. R., Srinivas Shakkottai, I-Hong Hou, and P. R. Kumar in Proc. of ACM MobiHoc 2018 [57, 103].

To provide strict per-packet latency guarantees, numerous theoretical solutions have been proposed to accommodate *per-packet deadline constraints* in wireless scheduling. For example, [109] proposes a theoretical framework to study wireless scheduling with per-packet deadline constraints. In this framework, packets not delivered on time are dropped. Later on, this framework has been applied to many other scenarios, such as utility maximization [110] and scheduling for both latency-constrained and best-effort traffic [111]. The performances of these protocols are usually measured by *timely-throughput*, i.e. the time average of the amount of data delivered within their deadlines. While the above solutions are promising, these theoretical results are unitless, and there has been no implementation for these ultra-low-latency wireless protocols. Therefore it is still unclear what is the minimum achievable latency and what level of timely-throughput can be obtained in practice.

This research aims to bridge the gap between theory and implementation for heterogeneous wireless networks supporting flows with strict per-packet latency constraints (real-time flows), as well as flows that have no latency constraints (non-real-time flows), while maintaining high overall system throughput. We propose PULS, a processor-supported software-defined wireless platfrom that can support ultra-low-latency scheduling protocols. The PULS platform consists of a host machine that has significant computational power in the form of a general-purpose multicore CPU, coupled with an software-defined radio (SDR) platform with FPGAs for low-level processing. PULS aims to leverage the higher clock speeds, and memory available on the Host machine (which are at least an order of magnitude higher than what is available on SDRs) for performing complex scheduling algorithms, while leveraging the deterministic performance of the FPGA while performing simple repetitive tasks associated with PHY and low-level MAC layers.

To achieve the required per-packet latency performance while performing schedul-

ing on the Host, PULS needs to address the following challenges:

1. **Low interfacing latency between software host and hardware.** There are three major factors that affect the end-to-end latency: (i) queuing delay on software host, (ii) interfacing latency between software host and hardware, and (iii) hardware processing time. Queuing delay depends mainly on the scheduling policy, and hardware processing time can usually be made small due to the high clock rate supported by current technology. Therefore, with a proper choice of scheduling policy and hardware component, interfacing latency between software host and hardware needs to be minimized in order to achieve ultra-low latency. In Chapter 5.4, we present a simple experiment that demonstrates the interfacing latency of PULS is indeed small compared to packet deadlines.

2. **Enforce per-packet deadline on a software-defined wireless platform.** PULS aims to support per-packet latency as low as 1 ms. When packets arrive at software host, they are first queued and start waiting for transmission according to some scheduling policy. The deadline of each packet in the queue needs to be tracked and checked before transmission. A packet that misses its deadline should be dropped from the queue. Moreover, due to the nature of SDRs, packet transmission is carried out on hardware while packet scheduling is often done on software host.

3. **Achieve realistic per-flow timely-throughput and overall system throughput.** Ultra-low latency needs to come with realistic timely-throughput. Given the same physical data rate, the overhead of enforcing per-packet deadlines could be quantified by the difference in total MAC-layer throughput between the networks with and without packet deadlines. However, this should not come at the cost of reduced network throughput. In Chapter 5.5, through an experimental study on

the achievable throughput, we show that PULS achieves almost the same MAC-layer throughput as that with no deadlines.

4. **Support functions working on heterogeneous time scales.** MAC layer functions operate on very different time scales. For example, an ACK response needs to be done within tens of microseconds. The transmission time of a typical data packet is between 0.5 to 1 ms. The target per-packet deadline is between 1 to 10 ms. The parameters of wireless protocols usually change over a period of at least several seconds to several minutes. In Chapter 5.3, we describe the separation principles of PULS which inherently incorporates the heterogeneity in time scale.

5. **Support various ultra-low-latency downlink applications.** For example, VR requires latency as low as 1 ms with moderate timely-throughput while factory automation needs ultra-low packet loss rate with latency of about 5-10 ms. PULS is able to support applications with totally different performance requirements and provide a programmable environment for different wireless protocols.

The main contributions can be summarized as follows:

- Through careful architectural deisgn, we develop a software-defined wireless testbed that supports low-latency per-packet scheduling.
- Through experiments, we show that a per-packet round-trip latency below 1 millisecond can be achieved on the proposed testbed.
- We implement real-time wireless scheduling algorithms on the proposed testbed and demonstrate the achievable total and per-flow timely-throughput with per-packet deadlines at millisecond level.

5.2 Related work

Various SDR architectures have been proposed to mitigate the interfacing overhead between software host and hardware. Just to name a few, [112] proposes a split-functionality framework to significantly reduce the communication overhead between software host and hardware. Similarly, [113] introduces Decomposable Medium Access Control (MAC) Framework to identify basic functional components according to both timeliness and degree of code reuse. However, neither of them considers per-packet deadline constraints nor provides any experimental results for ultra-low-latency wireless networks.

Most of the existing experimental studies for wireless LANs focus primarily on maximizing system throughput or throughput-based network utility. For example, to achieve maximium throughput, the well-known backpressure algorithm has been tailored and implemented for various scenarios, such as multi-hop wireless networks [114], TDMA-based MAC protocol [115] and wireless networks with intermittent connectivity [116]. Besides, for wireless LAN with random access, [117] implements an enhanced version of 802.11 DCF and demonstrates that it achieves near-optimal throughput as well as fairness with the original DCF. However, all of the above studies provide no support for packets with latency constraints. To address latency requirement for industrial control applications, RT-WiFi, a WiFi-compatible TDMA-based protocol, has been proposed and implemented on commercial 802.11 interface cards [118]. However, it cannot achieve both ultra-low latency and satisfactory timely-throughput performance for each user at the same time due to the nature of TDMA.

On the cellular side, several preliminary studies about 5G provide candidate solutions to enhance the low-latency capability via either numerical and experimental evaluation. [119] studies the trade-off between latency budget and required band-

width by applying the conventional OFDM framework to 5G networks through numerical analysis. However, these numerical results do not take the possible signaling and processing overhead into account. [120] provides experimental study for latency performance of 5G millimeter-wave networks with beamforming. However, this solution relies heavily on the beam-tracking technique and frame structure employed by cellular networks and cannot be directly applied to wireless LAN applications. Besides, [121] demonstrates a wireless testbed that is potentially capable of supporting millisecond-level end-to-end latency requirement. However, it supports only single link and does not take wireless scheduling issue into account.

5.3 Overview of the Testbed

In this section, we provide an overview of the design and main features of the proposed wireless testbed.

5.3.1 Hardware and Software

In PULS, each wireless node is a FPGA-based SDR from National Instruments (NI), which consists of a USRP-2953R that is connected through PCIe to a multi-core Windows laptop as the Host machine. The USRP-2953R is programmed through NI's LabVIEW Communication Systems Design Suite [122]. We start with the NI's 802.11 Application Framework (AF) [123], which provides the basic PHY and MAC functionality of the random-access-based 802.11 protocol without wireless scheduling features. Using the FPGA modules in 802.11 AF, we are able to modify the FPGA code to obtain a radio with a standard compliant PHY layer and a reconfigurable MAC layer. On the other hand, most of the wireless scheduling functions are implemented on the Host and will be discussed later in this section.

5.3.2 Architectural Design

The proposed testbed follows three major design principles.

- **Basic MAC Functions for Wireless Scheduling**: We are particularly interested in the implementation of low-latency centralized scheduling algorithms. To obtain a deployment that is both lightweight and capable of prototpying low-latency scheduling algorithms, we propose to borrow basic MAC functions from the random-access-based WiFi protocol. For example, we use the same slot time, interframe spacing timing (DIFS and SIFS), carrier sensing, and MAC layer acknowledgements as WiFi.

- **Mechanism-Policy Separation**: Mechanisms are functions or hardware blocks used to handle the low-level operations of packet transmissions over the network, whereas policy refers to the high-level specification of the scheduling algorithm. Each mechanism has a set of inputs, outputs, events, conditions to check and possible actions that can be performed. On the other hand, a policy specifies the set of enabling functions, the parameters for the conditions, the set of update functions and the transition relations for the state machine of the scheduling protocol. This mechanism-policy separation builds on the framework of Wireless MAC Processors, introduced by Tinnirello et al. [124].

- **Flexible MAC through Host-FPGA Separation**: For flexible MAC scheduling decisions, we employ a Host-FPGA separation, where high-level MAC functions such as packet scheduling and packet dropping, are implemented on the Host machine, and low-level and time-critical functions such as packet encoding/decoding, carrier sensing, ACK processing, and CRC checking are located on the FPGA. This design allows easy changes in packet scheduling decisions,

while still being able to achieve the required latency requirements. To push packets from the Host machine to FPGA, we define the Interface Communication Protocol (ICP), which specifies the format of the ICP header for Host packets, to convert Host packets into the actual MAC-layer MPDUs in FPGA.

5.3.3 Major Modules for Real-Time Wireless Scheduling

In PULS, scheduling is performed and repeated on a per-packet basis and is achieved by enforcing a sequence of mechanisms. Starting with the National Instruments 802.11 Application Framework (AF), we implement additional mechanisms on the Host machine as well as an FPGA for supporting real-time wireless networks. All scheduling-related mechanism blocks are implemented on the Host. We hightlight the major real-time wireless mechanisms implemented on the proposed testbed in Figure 5.1.

- **Prepend packet deadines**: Each real-time packet is prepended upon arrival with a Host timestamp, which indicates its absolute deadline. The time resolution of the timestamp can be as small as 1 microsecond. Note that the packets of the same flow are allowed to have different relative deadlines if required.
- **Deadline checking and packet dropping**: A packet will be dropped when it is not delivered by its deadline or it reaches its retransmission limit. Ideally, deadline checking and packet dropping shall be enforced in FPGA to keep track of the freshness of each packet until it gets delivered or dropped. Despite this, in PULS, deadline checking and packet dropping are implemented on Host instead of FPGA due to the following two reasons: (i) Due to the clock-driven logic and the fact that each packet is processed byte by byte in FPGA, packet dropping could incur significant overhead in FPGA. By contrast, the Host machine is able to drop packets in a much shorter period of time due to its rich computing

resource. (ii) The interfacing latency between the Host and FPGA, which will be discussed with details in Chapter 5.4, is nearly deterministic and relatively small compared to the per-packet deadline. Therefore, the absolute deadline pf each packet can be configured and checked either on the Host machine or the FPGA.

- **Update per-flow states**: Each scheduling policy requires continual update of state variables, such as deficits and queue lengths. To update state variables, the Host machine may also read register values in FPGA, such as the number of ACK and ACK timeout, through the PCIe interface.
- **Flow scheduler**: This is a generic module with per-flow state variables as input and a flow identifier as output. Moreover, it can support scheduling policy change on-the-fly. Since scheduling block is located on the Host machine, it is capable of supporting scheduling algorithms with high complexity.
- **Retransmissions**: Since wireless transmissions are lossy, we might need to retransmit the same packet for several times before delivering it in order to achieve timely-throughput guarantees. Retransmission is achieved in the following manner: a separate retransmission queue is created to store the duplicate of the current scheduled packet. The retransmission queue is assigned strictly higher priority than all the other normal flow queues, and therefore there is at most one packet in the retransmission queue at any time. The packet in the retransmission queue will get removed if it gets expired.
- **Prepare ICP packets**: Based on the ICP fromat, the scheduled packet is prepended with the corresponding ICP header, which consists of the configuration of bandwidth, target power level, modulation and coding scheme, source address, and destination addresses.

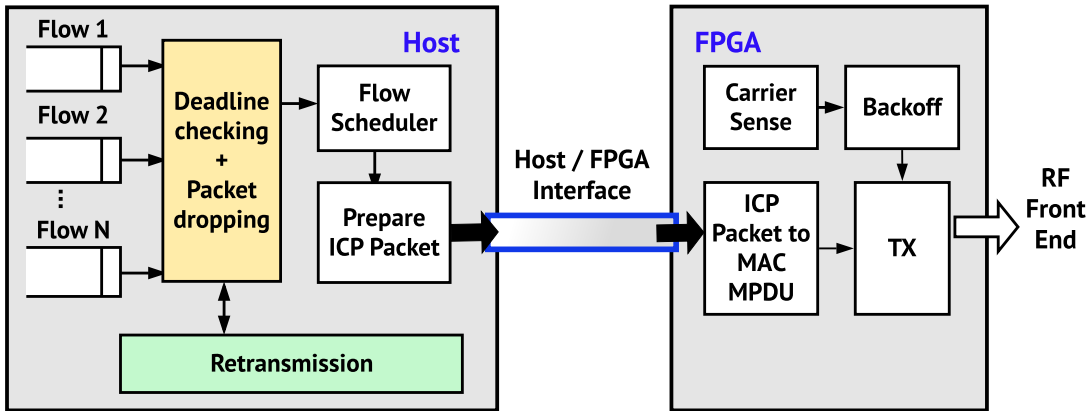


Figure 5.1: Major modules on PULS.

Based on the above mechanisms, the packet processing procedure can be summarized in Figure 5.2. First, the packet dropping mechanism scans the head-of-line packet of each queue, and drops the packets that are expired. This mechanism only requires the references of each flow queue as well as the current Host timestamp as input arguments. It also has an output to inform other mechanisms whether or not a packet has been dropped. Next, we have a mechanism that updates the state of the flows. For example, under the scheduling policy proposed in [109], the deficit associated with each flow is updated based on whether or not an ACK was received and if a packet has been dropped from the queue. Lastly, the flow scheduler decides which queue to schedule based on the current states of the flows.

5.4 Supporting Sub-Millisecond Per Packet Latency

In this section, we describe an experiment to measure the round-trip time of a packet transmission.

There are three major factors that affect the round-trip time: (i) queuing delay on software host, (ii) interfacing latency between software host and hard- ware, and (iii)

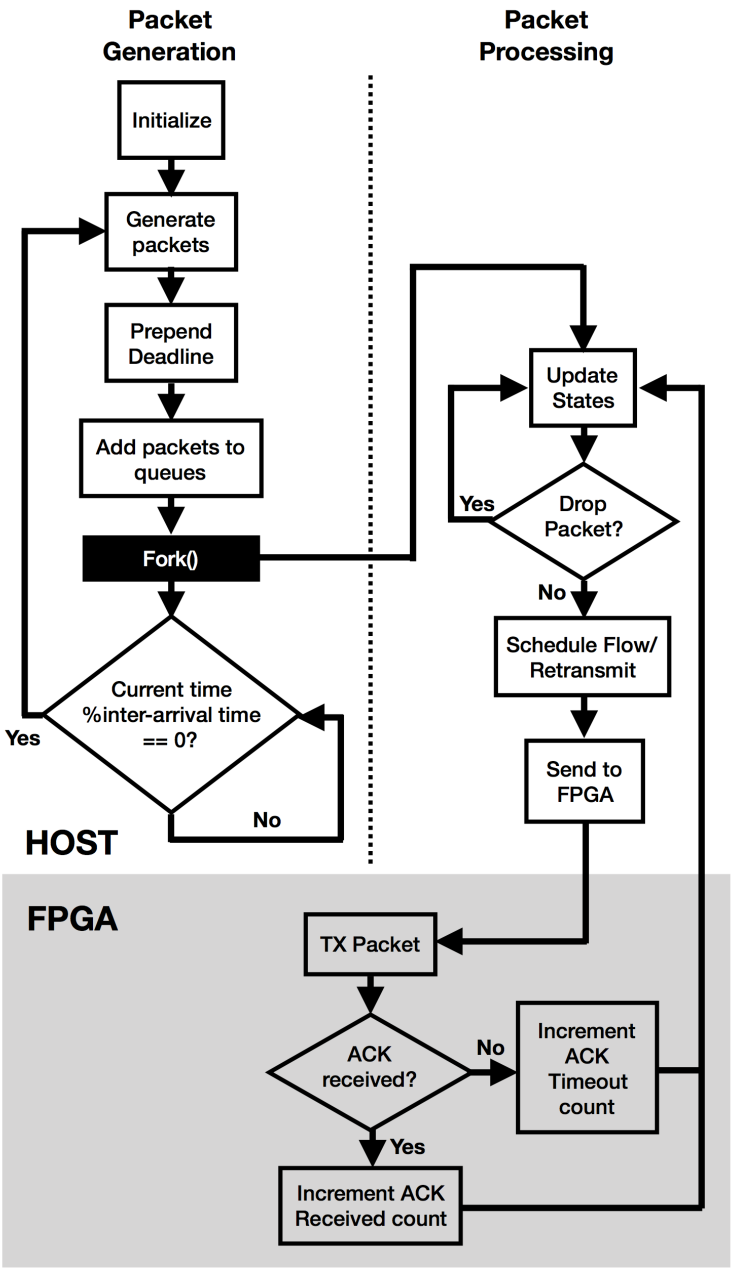


Figure 5.2: Packet transmission procedure of PULS.

hardware processing time. Queuing delay depends mainly on the scheduling policy, and hardware processing time can usually be made small due to the high clock rate supported by current technology. Therefore, with a proper choice of scheduling policy and hardware component, interfacing latency between software host and hardware needs to be minimized in order to achieve low latency. The experiment can be summarized as follows:

1. Test packets of fixed payload size arrive at the Host periodically. The period is set to be large enough such that at each time there is only 1 test packet waiting for transmission in the Host queue. This completely eliminates the effect of queueing delay in the Host.

2. When a test packet arrives at the Host, it is given a timestamp denoted by t_h (read by the Host from an FPGA register) and then forwarded to the FPGA through an interfacing channel immediately.

3. When FPGA detects the new test packet, FPGA starts processing the ICP header and retrieves t_h from the header. Along with the current FPGA counter denoted by t_f, the Host-to-FPGA interfacing latency can be derived as $t_f - t_h$.

4. The packet is then transmitted. When the corresponding ACK is received, FPGA reads the current timestamp t_r and calculates the round-trip latency as $t_r - t_h$.

In the experiments, we measure both latency metrics for 50000 packets with a fixed interarrival time of 10ms.

Figure 5.3(a) shows the empirical cumulative distribution function (CDF) of the Host-to-FPGA interfacing latency for packets with 1500 bytes payload at a data rate of 54Mbps. The mean interfacing latency is around 192 μs, and the 90, 95, and 99 percentiles are 233.4, 257.6, and 316.1 μs, respectively. Table 5.1 further summarizes

Table 5.1: Host-to-FPGA latency results

Data rate (Mbps)	Payload size (bytes)	Host-to-FPGA latency (μs)			
		Mean	90%	95%	99%
54	500	185.2	227.6	251.5	301.8
54	1000	189.7	235	259	315
54	1500	192.2	233.4	257.6	316.1
24	500	187.8	228.4	252.7	305.7
24	1000	188.3	230.6	254.3	304
24	1500	189.2	231.5	255	304.6

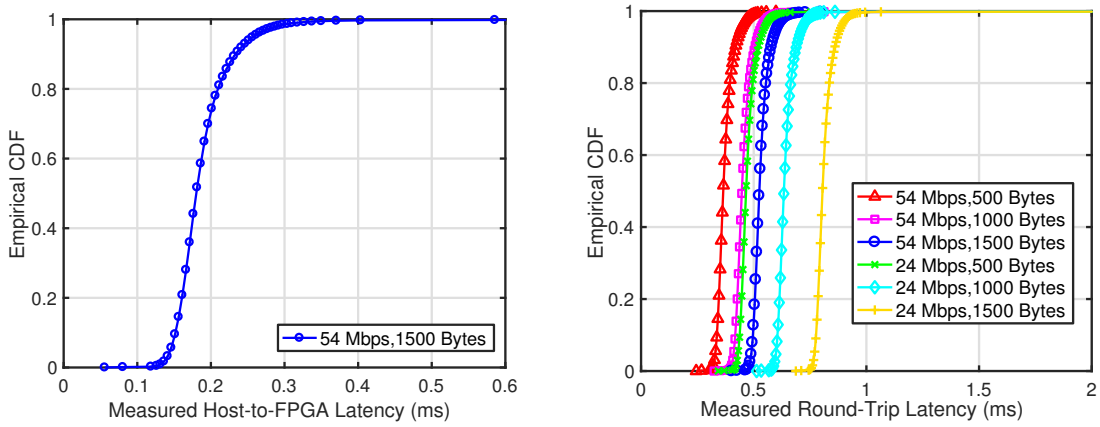
Table 5.2: Round-trip latency results

Data rate (Mbps)	Payload size (bytes)	Round-trip latency (μs)			
		Mean	90%	95%	99%
54	500	377.0	419.2	443.4	494.4
54	1000	459.9	505.1	529.2	585.0
54	1500	536.9	578.5	602.3	660.8
24	500	479.5	520.2	544.4	598.1
24	1000	646.6	688.9	712.7	762.8
24	1500	817.9	860.2	883.9	933.7

the statistics of the interfacing latency for different data rates and payload sizes. We see that both the mean and the percentiles of the Host-to-FPGA latency are almost invariant, regardless of data rate and payload size. Therefore, the proposed wireless testbed indeed exhibits low and predictable interfacing latency.

Next, Figure 5.3(b) shows the empirical CDF of round-trip latency, and Table 5.1 summarizes the statistics of round-trip latency for different data rates and payload sizes. Since round-trip latency consists of both transmission time and Host-to-FPGA interfacing latency, it varies with the physical data rate and the payload size. For the six test cases listed in Table 5.1, the maximum 99 percentile round-trip latency is 933.7 μs. Therefore, the testbed is indeed able to guarantee a round-trip latency

below 1ms with high probability even for large packet sizes and moderate physical data rates.



(a) Empirical CDF of Host-to-FPGA latency for payload size = 1500 bytes and data rate = 54Mbps.

(b) Empirical CDFs of round-trip latency for various data rates and payload sizes.

Figure 5.3: Empirical CDF of the interfacing latency and round-trip latency.

5.5 Supporting Low-Latency Wireless Networking

In this section, we demonstrate the capability of the proposed testbed to support real-time wireless scheduling algorithms.

5.5.1 Experimental Setup

In the following experiments, we consider a wireless network with one AP serving up to two downlink clients. Each client is associated with one real-time flow with per-packet deadlines as well as one non-real-time flow without deadline constraints. Each real-time flow has a delivery ratio requirement between 0 and 1, which represents the minimum fraction of packets required to be delivered by the deadlines. We implement the Largest-Deficit-First (LDF) policy [109], which has been shown to

achieve optimal timely-throughput for real-time wireless networks, as well as three baseline policies: Longest Queue First (LQF), Round Robin (RR), and Randomized (Random) policy.

- LDF policy: Based on [109], the deficit of each real-time flow is defined as the product of delivery ratio requirement and the number of arrivals subtracted by the number of delivered packets. Under LDF, the AP schedules the real-time flow with the largest deficit (ties broken arbitrarily) and selects the non-real-time flow with the largest queue length if all of the real-time flows have empty queues.

- LQF policy: Under LQF, the AP does not distinguish the real-time flows from non-real-time flows. Instead, the AP simply schedules the flow with the largest queue length.

- RR policy: Under RR, the AP schedules flows in the following order, i.e. real-time flow for client 1, real-time flow for client 2, non-real-time flow for client 1, and then non-real-time flow for client 2. If the designated queue is empty, the AP will schedule the next queue.

- Random policy: Under Random, the AP selects one flow uniformly in random among those with an non-empty queue.

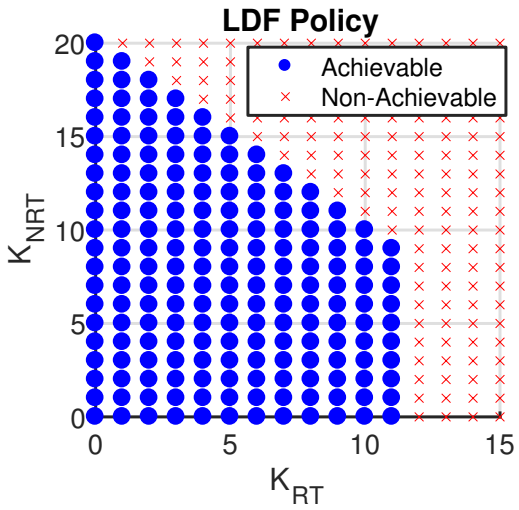
To avoid transmitting expired packets of real-time flows, it is more efficient to drop expired packets as suggested in [109]. However, the feature of dropping expired packets is currently not included in most implementations. To ensure a fair comparison, we enable packet dropping for all of the four policies. The experiments are run using 1500B packets and IEEE 802.11a with a 54Mbps physical data rate.

5.5.2 Achievable Throughput Regions

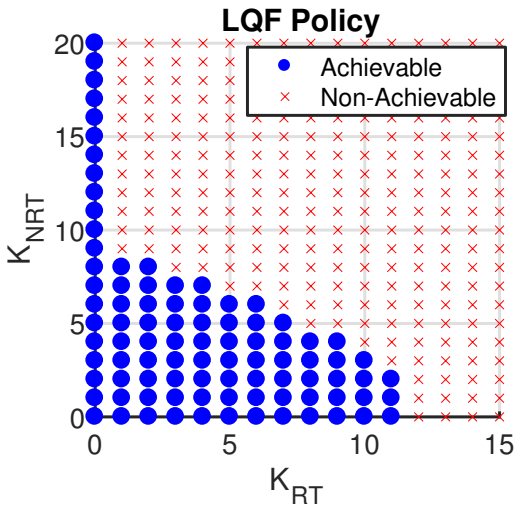
In this section, we empirically characterize the *achievable throughput region* under LDF policy as well as the other three baseline policies. The achievable throughput region under a policy is defined as the arrival rate region where both the total deficit (accumulated when packets are dropped) and the queue length stay finite for any period of time. For ease of exposition, we study the case where the AP serves only one client with one real-time flow and one non-real-time flow. We consider the following arrival process: packets arrive in batch every 5 ms and the number of packets in each batch are uniformly distributed from 0 to K_{RT} for real-time flows, and K_{NRT} for non-real-time flows. The per-packet deadlines are chosen to be 5 ms with the delivery ratio requirement equal to 0.99.

Figure 5.4(a)-5.4(d) show the achievable throughput region under the four policies. First, LDF policy achieves an achievable throughput region that is strictly larger than that under the other three baseline policies. LDF always schedules real-time flows first, so the drop off in the achievable throughput region is linear until $K_{RT} = 11$, after which the deficit starts increasing indefinitely for this delivery ratio and deadline. While Random and RR have similar achievable throughput regions, RR performs slightly better than Random when K_{RT} is small (less than 6) since the interservice time is more regularized under RR than Random. Due to the service irregularity under Random, there is more idle time (when there is no packet to transmit) near the end of each deadline under Random than unedr RR policy when K_{RT} is small. LQF performs better for higher value of K_{RT} than Random and RR since the real-time flow get scheduled more frequently than the non-real-time flow under LQF. However, LDF suffers significantly when K_{NRT} is larger than K_{RT} since the non-real-time flow gets scheduled first. In this scenario, all policies support up to a

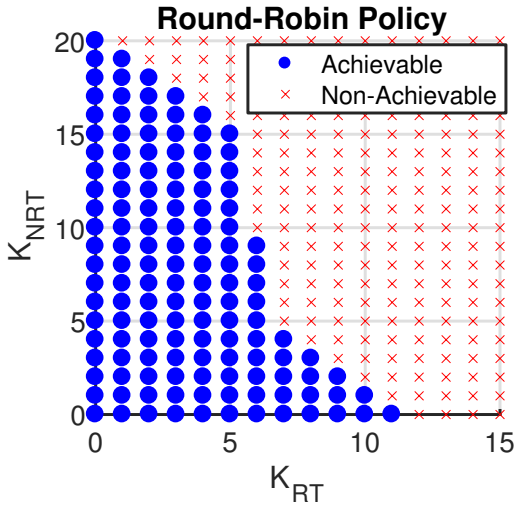
$K_{RT} = 11$ and serve up to 20 packets in total every 5 ms.



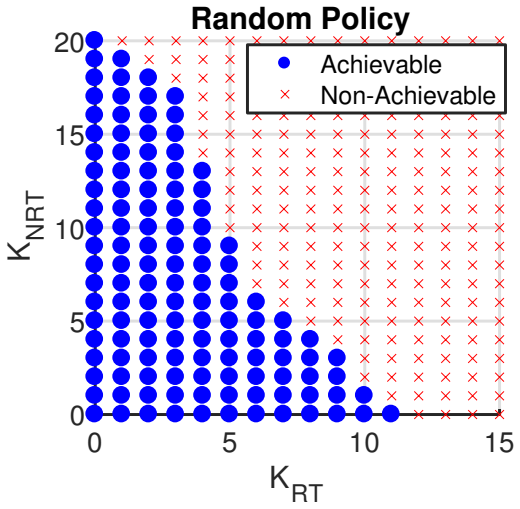
(a) Throughputs for K_{RT}=4 and K_{NRT}=4.



(b) Throughputs for K_{RT}=3 and K_{NRT}=6.



(c) Throughputs for K_{RT}=3, K_{NRT}=5 for client 1, K_{RT1}=4, K_{NRT}=6 for client 2.



(d) Throughputs for K_{RT}=7, K_{NRT}=4 for client 1, K_{RT1}=3, K_{NRT}=5 for client 2.

Figure 5.4: Achievable throughput regions under the four scheduling policies.

5.5.3 Network Throughput and Loss Ratio Performance

In this section, we study throughput and loss ratio performance of a network formed by one AP and two clients. As in Chapter 5.5.2, packets still arrive in batch every 5 ms and the number of packets in each batch are uniformly distributed from 0 to K_{RT} for real-time flows, and K_{NRT} for non-real-time flows. We study both symmetric and asymmetric cases, and the corresponding network throughput and loss ratio performance are shown in Figure 5.5(a)-5.5(d) and Table 5.3. We discuss the results of each case as follows:

Table 5.3: Loss Ratios of Real-Time Flows

Arrival Rate	Deadline (ms)		Delivery Ratio		Policy	Loss Ratio (%)	
	C1	C2	C1	C2		C1	C2
$K_{RT1}=4$ $K_{NRT1}=4$ $K_{RT2}=4$ $K_{NRT2}=4$	3	3	0.98	0.98	LDF	<2	<2
					LQF	23.51	17.12
					Rand	15.17	15.62
					RR	15.07	16.03
$K_{RT1}=3$ $K_{NRT1}=6$ $K_{RT2}=3$ $K_{NRT2}=6$	2	2	0.95	0.95	LDF	<5	<5
					LQF	60.24	54.97
					Rand	26.48	27.22
					RR	19.96	21.27
$K_{RT1}=3$ $K_{NRT1}=5$ $K_{RT2}=4$ $K_{NRT2}=6$	2	3	0.97	0.98	LDF	<3	<2
					LQF	61.45	27.35
					Rand	27.2	18.07
					RR	20.09	17.13
$K_{RT1}=7$ $K_{NRT1}=4$ $K_{RT2}=3$ $K_{NRT2}=5$	5	2	0.98	0.99	LDF	<2	<1
					LQF	7.18	63.51
					Rand	17.01	29.46
					RR	19.14	21.42

- **Case 1**: In the fully-symmetric case, we set $K_{RT}=4$ and $K_{NRT}=4$, the per-packet deadline of the real-time flows to 3 ms with a delivery ratio requirement

of 0.98 for both clients. As shown in Figure 5.5(a), we see that while most protocols perform relatively well, LDF achieves a higher timely-throughput and overall throughput for both clients. Moreover, based on Table 5.3, we observe that LDF achieves the required delivery ratio for both clients, while the other three policies have about 7 to 12 times more packet expiries.

- **Case 2**: We then consider the case where $K_{RT} = 3$, $K_{NRT} = 6$, and a per-packet deadline of 2 ms with delivery ratio requirement equal to 0.95 for both clients. Similar to Case 1, under such stringent per-packet deadline, LDF still achieves the required delivery ratio and strictly better overall throughput than the other three policies.

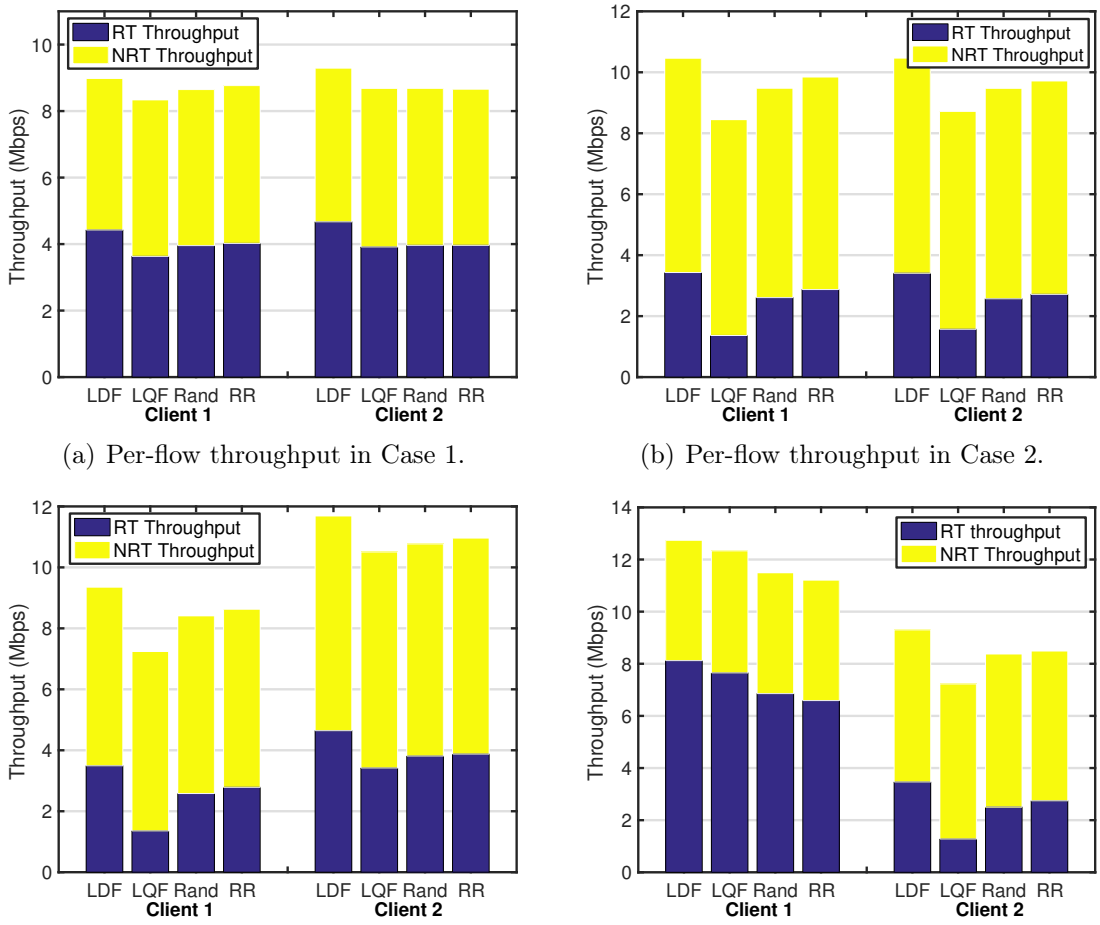
- **Case 3**: In the asymmetrical scenarios, we consider the following two sets of parameters.

  - **Case 3-A**: $K_{RT} = 3$ and $K_{NRT} = 5$ for client 1, $K_{RT} = 4$ and $K_{NRT} = 6$ for client 2. Client 1's deadline is 2 ms with 0.97 delivery ratio, and client 2's deadline is 3ms with 0.98 delivery ratio.

  - **Case 3-B**: $K_{RT} = 7$ and $K_{NRT} = 4$ for client 1, $K_{RT} = 3$ and $K_{NRT} = 5$ for client 2. Client 1's deadline is 5 ms with 0.98 delivery ratio, and client 2's deadline is 2ms with 0.99 delivery ratio.

  Similar to Case 1 and Case 2, in these heterogeneous scenarios, LDF still achieves the required delivery ratio and much better overall throughput (roughly 22 Mbps in total) than the other three policies. Also note that with NI's SDR under 802.11a with a physical data rate of 54Mbps and a packet size of 1500B, the maximum MAC-layer throughput for a single link is no larger than 30 Mbps [123]. In other words, the testbed still achieves more than 70% of the link throughput after in-

tegrating all the real-time scheduling functionality. Therefore, we show that the proposed testbed can indeed support realistic total and per-flow throughputs with strict per-packet deadlines.



(a) Per-flow throughput in Case 1.

(b) Per-flow throughput in Case 2.

(c) Per-flow throughput in Case 3-A.

(d) Per-flow throughput in Case 3-B.

Figure 5.5: Throughput performance under four different sets of arrival rates.

5.6 Lessons Learned From Implementation

In this section, we discuss about the lessons learned from implementation and how to further improve the performance of the proposed testbed. As mentioned

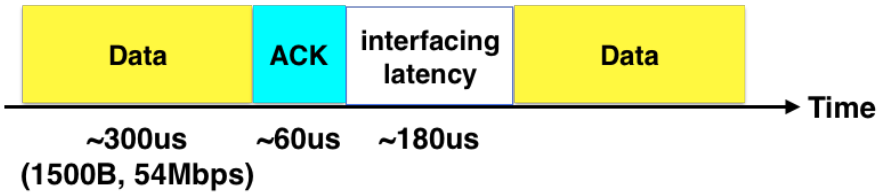


Figure 5.6: The interfacing latency in a packet transmission.

in Chapter 5.5.3, there is about 30% throughput loss when the real-time wireless features are included. The throughput loss results mainly from the Host-to-FPGA interfacing latency. Figure 5.6 shows that the typical timeline of packet transmissions on PULS . Under a data rate of 54 Mbps and 1500B packets, the actual air time of a packet transmission is about 300 us. Moreover, the air time of an ACK plus the required SIFS time is about 60 us. Recall that the average interfacing latency is about 180 us based on the results in Chapter 5.4. Therefore, the effective air time of each packet transmission is about 540 us while the actual air time of a data packet and the corresponding ACK takes only about 360 us. Hence, the interfacing latency is the bottleneck of system throughput and needs to be reduced.

To improve system throughput, we suggest four possible solutions to reducing the interfacing latency as follows:

1. **(Hardware) Use FPGAs with embedded processors.** The interfacing latency depends heavily on the thread scheduling of the processor, which allocates computing resource to multiple applications running simultaneously. Based on this observation, we may use FPGAs with embedded processors, which is capable of achieving lower interrupt latency than that of a Host laptop.

2. **(Architecture) Move the flow scheduler from Host to FPGA.** One could completely obviate the Host-to-FPGA interfacing latency by implement-

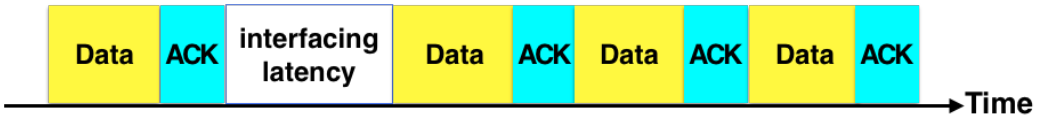


Figure 5.7: An example of look-ahead packet scheduling.

ing the flow scheduler in FPGA. However, as mentioned in Chapter 5.3.3, it requires much more design efforts to handle deadline checking and packet dropping in FPGA.

3. **(Algorithm) Use batch scheduling policies.** In PULS, the interfacing latency takes place before each packet transmission due to the need of per-packet scheduling. To amortize the interfacing latency, we may consider batch scheduling policies as shown in Figure 5.6. By allowing scheduling for the next K consecutive transmissions, the effective interfacing latency would be reduced by K times. This manifests the tradeoff between the effective interfacing latency and the schedulign design.

5.7 Summary

This chapter presents a software-defined wireless testbed for implementing low-latency wireless scheduling algorithms. Based on the careful architectural design, we show via extensive experiments that the testbed achieves a per-packet round-trip latency below 1 millisecond as well as realistic timely-throughput with strict per-packet deadlines.

6. CONCLUSIONS AND FUTURE RESEARCH DIRECTIONS

In this dissertation, we aim to explore the challenges and opportunities of the emerging IoT technology by designing optimal network algorithms. We have studied three important IoT applications and proposed network algorithms with theoretical guarantees for each application. We conclude by summarizing the key results and providing promising directions for future research.

- In Chapter 2, we study the scheduling problem for multi-hop networked transportation systems and propose the Biased Max-Pressure policy that is throughput-optimal with non-zero switch-over delay. While we focus on the case where the queue capacity is infinite, we show via simulation that the proposed policy performs well with finite queues. In [125], it has been shown that a Max-Weight-like policy is throughput-optimal for finite queues without switch-over delay. Designing a throughput-optimal policy for multi-hop networks with both finite queues and non-zero switch-over delay is still an open problem.

- In Chapter 3, we study the scheduling problem for wireless on-demand video delivery and propose QoE-optimal scheduling policies by applying diffusion approximation. The proposed policy is evaluated via extensive simulation and experiments on a software-defined testbed. Extensions to diffusion approximation for live video streaming is an interesting direction for future research.

- In Chapter 4, we address decentralized medium access for real-time wireless ad hoc networks. We design a generic protocol by using only collision-free backoff and carrier sensing. Extensions to massive IoT connecticvity with strict per-packet deadlines is a promising direction to explore.

- In Chapter 5, we present a software-defined wireless testbed that achieves ultra-low per-packet latency and is able to support real-time wireless networks with strict per-packet deadlines. As mentioned in Chapter 5.6, we expect to further reduce the interfacing latency and improves the network throughput for the emerging high-throughput ultra-low latency applications.

REFERENCES

[1] P.-C. Hsieh, X. Liu, J. Jiao, I.-H. Hou, Y. Zhang, and P. R. Kumar, "Throughput-Optimal Scheduling for Multi-Hop Networked Transportation Systems With Switch-Over Delay," in *Proc. of ACM MobiHoc*, 2017.

[2] K. Tiaprasert, Y. Zhang, X. B. Wang, and X. Zeng, "Queue length estimation using connected vehicle technology for adaptive signal control," *IEEE Trans. Intell. Transp. Syst.*, vol. 16, no. 4, pp. 2129–2140, 2015.

[3] B. E. Badillo, H. Rakha, T. W. Rioux, and M. Abrams, "Queue length estimation using conventional vehicle detector and probe vehicle data," in *Proc. of Intelligent Transportation Systems (ITSC)*, pp. 1674–1681, IEEE, 2012.

[4] L. Tassiulas and A. Ephremides, "Stability properties of constrained queueing systems and scheduling policies for maximum throughput in multihop radio networks," *IEEE Transactions on Automatic Control*, vol. 37, no. 12, pp. 1936–1948, 1992.

[5] P. Varaiya, "Max pressure control of a network of signalized intersections," *Transp. Res. Part C: Emerg. Technol.*, vol. 36, pp. 177–195, 2013.

[6] T. Wongpiromsarn, T. Uthaicharoenpong, Y. Wang, E. Frazzoli, and D. Wang, "Distributed traffic signal control for maximum network throughput," in *Proc. of IEEE ITSC*, pp. 588–595, 2012.

[7] N. Xiao, E. Frazzoli, Y. Li, Y. Wang, and D. Wang, "Pressure releasing policy in traffic signal control with finite queue capacities," in *Proc. of IEEE CDC*, pp. 6492–6497, IEEE, 2014.

[8] J. Gregoire, E. Frazzoli, A. de La Fortelle, and T. Wongpiromsarn, "Back-pressure traffic signal control with unknown routing rates," *IFAC Proceedings Volumes*, vol. 47, no. 3, pp. 11332–11337, 2014.

[9] S. Lämmer and D. Helbing, "Self-control of traffic lights and vehicle flows in urban road networks," *J. Stat. Mech: Theory Exp.*, vol. 2008, no. 04, p. P04019, 2008.

[10] A. Ghavami, K. Kar, and S. Ukkusuri, "Delay analysis of signal control policies for an isolated intersection," in *Proc. of IEEE ITSC*, pp. 397–402, 2012.

[11] X. Liu, K. Ma, and P. Kumar, "Towards provably safe mixed transportation systems with human-driven and automated vehicles," in *Proc. of IEEE CDC*, pp. 4688–4694, 2015.

[12] P. R. Lowrie, "The Sydney coordinated adaptive traffic system-principles, methodology, algorithms," in *Proc. of International Conference on Road Traffic Signalling*, no. 207, 1982.

[13] SCOOT, "Scoot - the world's leading adaptive traffic control system," 2014.

[14] M. Armony and N. Bambos, "Queueing dynamics and maximal throughput scheduling in switched processing systems," *Queueing Syst.*, vol. 44, no. 3, pp. 209–252, 2003.

[15] Y.-C. Hung and C.-C. Chang, "Dynamic scheduling for switched processing systems with substantial service-mode switching times," *Queueing Syst.*, vol. 60, no. 1-2, pp. 87–109, 2008.

[16] C. W. Chan, M. Armony, and N. Bambos, "Maximum Weight Matching with Hysteresis in Overloaded Queues with Setups," *Queueing Syst. Theory Appl.*, vol. 82, pp. 315–351, Apr 2016.

[17] G. Celik, S. C. Borst, P. A. Whiting, and E. Modiano, "Dynamic Scheduling with Reconfiguration Delays," *Queueing Syst. Theory Appl.*, vol. 83, pp. 87–129, Jun 2016.

[18] Transportation Research Board, *Highway capacity manual*. National Research Council, 2000.

[19] H. Chen and D. Yao, *Fundamentals of Queueing Networks: Performance, Asymptotics and Optimization*. Springer-Verlag, 2001.

[20] M. J. Neely, "Stochastic network optimization with application to communication and queueing systems," *Synthesis Lectures on Communication Networks*, vol. 3, no. 1, pp. 1–211, 2010.

[21] J. Gregoire, X. Qian, E. Frazzoli, A. De La Fortelle, and T. Wongpiromsarn, "Capacity-aware backpressure traffic signal control," *IEEE Trans. Control Netw. Syst*, vol. 2, no. 2, pp. 164–173, 2015.

[22] PTV VISSIM, "Transportation planning, traffic engineering and traffic simulation," 2017.

[23] Synchro Studio, "Synchro studio: Planning & analysis software," 2017.

[24] G. D. Celik and E. Modiano, "Scheduling in networks with time-varying channels and reconfiguration delay," *IEEE/ACM Trans. Netw.*, vol. 23, no. 1, pp. 99–113, 2015.

[25] P. C. Hsieh and I. H. Hou, "Heavy-traffic analysis of QoE optimality for on-demand video streams over fading channels," in *Proc. of IEEE INFOCOM*, pp. 1–9, April 2016.

[26] Cisco, "Cisco visual networking index: Global mobile data traffic forecast update, 2014-2019," Feb. 2015.

[27] N. Staelens, S. Moens, W. V. den Broeck, I. MarieÌĺn, B. Vermeulen, P. Lambert, R. V. de Walle, and P. Demeester, "Assessing quality of experience of IPTV and video on demand services in real-life environments," *IEEE Transactions on Broadcasting*, vol. 56, no. 4, pp. 458–466, 2010.

[28] R. Mok, E. Chan, and R. Chang, "Measuring the quality of experience of HTTP video streaming," in *Proc. of IFIP/IEEE International Symposium on Integrated Network Management (IM)*, pp. 485–492, 2011.

[29] F. Dobrian, V. Sekar, A. Awan, I. Stoica, D. Joseph, A. Ganjam, J. Zhan, and H. Zhang, "Understanding the impact of video quality on user engagement," in *Proc. of ACM SIGCOMM*, pp. 362–373, 2011.

[30] G. Liang and B. Liang, "Effect of delay and buffering on jitter-free streaming over random VBR channels," *IEEE Transactions on Multimedia*, vol. 10, pp. 1128–1141, Oct 2008.

[31] A. ParandehGheibi, M. Medard, A. Ozdaglar, and S. Shakkottai, "Avoiding interruptions - a QoE reliability function for streaming media applications," *IEEE Journal on Selected Areas in Communications*, vol. 29, no. 5, pp. 1064–1074, 2011.

[32] Y. Xu, E. Altman, R. El-Azouzi, M. Haddad, S. Elayoubi, and T. Jimenez, "Analysis of buffer starvation with application to objective QoE optimization of streaming services," *IEEE Transactions on Multimedia*, vol. 16, no. 3, pp. 813–827, 2014.

[33] A. Anttonen and A. MammelaÌĹ, "Interruption probability of wireless video streaming with limited video lengths," *IEEE Transactions on Multimedia*, vol. 16, pp. 1176–1180, June 2014.

[34] J. Yang, H. Hu, H. Xi, and L. Hanzo, "Online buffer fullness estimation aided adaptive media playout for video streaming," *IEEE Transactions on Multimedia*, vol. 13, pp. 1141–1153, Oct 2011.

[35] T. Luan, L. Cai, and X. Shen, "Impact of network dynamics on user's video quality: Analytical framework and QoS provision," *IEEE Transactions on Multimedia*, vol. 12, pp. 64–78, Jan 2010.

[36] Y. Xu, S. Elayoubi, E. Altman, and R. El-Azouzi, "Impact of flow-level dynamics on QoE of video streaming in wireless networks," in *Proc. of IEEE INFOCOM*, pp. 2715–2723, 2013.

[37] Y. Xu, Z. Xiao, H. Feng, T. Yang, B. Hu, and Y. Zhou, "Modeling buffer starvations of video streaming in cellular networks with large-scale measurement of user behavior," *IEEE Transactions on Mobile Computing*, vol. 16, no. 8, pp. 2228–2245, 2017.

[38] D. De Vleeschauwer, H. Viswanathan, A. Beck, S. Benno, G. Li, and R. Miller, "Optimization of HTTP adaptive streaming over mobile cellular networks," in *Proc. of IEEE INFOCOM*, pp. 898–997, April 2013.

[39] P. Chandur and K. Sivalingam, "Quality of experience aware video scheduling in LTE networks," in *Proc. of NCC*, pp. 1–6, Feb 2014.

[40] R. Bhatia, T. Lakshman, A. Netravali, and K. Sabnani, "Improving mobile video streaming with link aware scheduling and client caches," in *Proc. of IEEE INFOCOM*, pp. 100–108, April 2014.

[41] K. S. Kim, C.-P. Li, and E. Modiano, "Scheduling multicast traffic with deadlines in wireless networks," in *Proc. of IEEE INFOCOM*, pp. 2193–2201, April 2014.

[42] M. Li, P. H. Tan, S. Sun, and Y. H. Chew, "QoE-aware scheduling for video streaming in 802.11 n/ac-based high user density networks," in *IEEE Vehicular Technology Conference (VTC Spring)*, pp. 1–5, 2016.

[43] S. Cicalo, N. Changuel, V. Tralli, B. Sayadi, F. Faucheux, and S. Kerboeuf, "Improving QoE and fairness in HTTP adaptive streaming over LTE network," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 26, no. 12, pp. 2284–2298, 2016.

[44] C. Li, H. Xiong, J. Zou, and D. O. Wu, "Joint dynamic rate control and transmission scheduling for scalable video multirate multicast over wireless networks," *IEEE Transactions on Multimedia*, vol. 20, no. 2, pp. 361–378, 2018.

[45] A. Anand and G. de Veciana, "Measurement-based scheduler for multi-class QoE optimization in wireless networks," in *Proc. of IEEE INFOCOM*, pp. 1–9, 2017.

[46] V. Joseph and G. de Veciana, "NOVA: QoE-driven optimization of DASH-based video delivery in networks," in *Proc. of IEEE INFOCOM*, pp. 82–90, April 2014.

[47] K. Xiao, S. Mao, and J. K. Tugnait, "QoE-driven resource allocation for DASH over OFDMA networks," in *Proc. of GLOBECOM*, pp. 1–6, 2016.

[48] I.-H. Hou and P.-C. Hsieh, "QoE-optimal scheduling for on-demand video streams over unreliable wireless networks," in *Proc. of ACM MobiHoc*, pp. 207–216, 2015.

[49] L. Tassiulas and A. Ephremides, "Dynamic server allocation to parallel queues with randomly varying connectivity," *IEEE Transactions on Information The-*

ory, vol. 39, no. 2, pp. 466–478, 1993.

[50] M. Neely, "Delay analysis for Max Weight opportunistic scheduling in wireless systems," *IEEE Transactions on Automatic Control*, vol. 54, pp. 2137–2150, Sept 2009.

[51] J. Harrison and M. Lopez, "Heavy traffic resource pooling in parallel-server systems," *Queueing Systems*, vol. 33, no. 4, pp. 339–368, 1999.

[52] J. M. Harrison, *Brownian motion and stochastic flow systems*. Wiley New York, 1985.

[53] M. J. Neely, E. Modiano, and C. E. Rohrs, "Dynamic power allocation and routing for time-varying wireless networks," *IEEE Journal on Selected Areas in Communications*, vol. 23, no. 1, pp. 89–103, 2005.

[54] H. Kushner and P. Whiting, "Convergence of proportional-fair sharing algorithms under general conditions," *IEEE Transactions on Wireless Communications*, vol. 3, pp. 1250–1259, July 2004.

[55] "Live encoder settings, bitrates and resolutions." `https://support.google.com/youtube/answer/2853702?hl=EN`.

[56] S. Yau, L. Ge, P.-C. Hsieh, I.-H. Hou, S. Cui, P. Kumar, A. Ekbal, and N. Kundargi, "WiMAC: Rapid Implementation Platform for User Definable MAC Protocols Through Separation," in *Proc. of ACM SIGCOMM (demo paper)*, pp. 109–110, 2015.

[57] S. Yau, P.-C. Hsieh, R. Bhattacharyya, K. Bhargav K. R., S. Shakkottai, I.-H. Hou, and P. R. Kumar, "PULS: Processor-Supported Ultra-Low Latency Scheduling," in *Proc. of ACM MobiHoc*, 2018.

[58] P.-C. Hsieh and I.-H. Hou, "A decentralized medium access protocol for real-time wireless ad hoc networks with unreliable transmissions," in *Proc. of IEEE ICDCS*, 2018.

[59] A. Dua and N. Bambos, "Downlink wireless packet scheduling with deadlines," *IEEE Transactions on Mobile Computing*, vol. 6, no. 12, 2007.

[60] I. H. Hou, V. Borkar, and P. R. Kumar, "A Theory of QoS for Wireless," in *Proc. of IEEE INFOCOM*, pp. 486–494, 2009.

[61] I.-H. Hou *et al.*, "Scheduling heterogeneous real-time traffic over fading wireless channels," *IEEE/ACM Transactions on Networking (TON)*, vol. 22, no. 5, pp. 1631–1644, 2014.

[62] K. S. Kim, C.-P. Li, and E. Modiano, "Scheduling multicast traffic with deadlines in wireless networks," in *Proc. of IEEE INFOCOM*, pp. 2193–2201, 2014.

[63] I.-H. Hou, "Broadcasting delay-constrained traffic over unreliable wireless links with network coding," *IEEE/ACM Transactions on Networking (TON)*, vol. 23, no. 3, pp. 728–740, 2015.

[64] J. J. Jaramillo and R. Srikant, "Optimal scheduling for fair resource allocation in ad hoc networks with elastic and inelastic traffic," in *Proc. of IEEE INFOCOM*, pp. 1–9, 2010.

[65] L. Deng, C.-C. Wang, M. Chen, and S. Zhao, "Timely wireless flows with general traffic patterns: Capacity region and scheduling algorithms," *IEEE/ACM Transactions on Networking (TON)*, 2017.

[66] J. J. Jaramillo, R. Srikant, and L. Ying, "Scheduling for optimal rate allocation in ad hoc networks with heterogeneous delay constraints," *IEEE Journal on Selected Areas in Communications*, vol. 29, no. 5, pp. 979–987, 2011.

[67] X. Kang, W. Wang, J. J. Jaramillo, and L. Ying, "On the performance of largest-deficit-first for scheduling real-time traffic in wireless networks," *IEEE/ACM Transactions on Networking*, vol. 24, no. 1, pp. 72–84, 2016.

[68] D. Camps-Mur, A. Garcia-Saavedra, and P. Serrano, "Device-to-device communications with Wi-Fi Direct: overview and experimentation," *IEEE Wireless Communications*, vol. 20, no. 3, pp. 96–104, 2013.

[69] J. Chen, X. Cao, P. Cheng, Y. Xiao, and Y. Sun, "Distributed collaborative control for industrial automation with wireless sensor and actuator networks," *IEEE Transactions on Industrial Electronics*, vol. 57, no. 12, pp. 4219–4230, 2010.

[70] L. Jiang and J. Walrand, "Convergence and stability of a distributed csma algorithm for maximal network throughput," in *Proc. of IEEE Conference on Decision and Control (CDC)*, pp. 4840–4845, 2009.

[71] S. Rajagopalan, D. Shah, and J. Shin, "Network adiabatic theorem: an efficient randomized protocol for contention resolution," in *Proc. of ACM SIGMETRICS performance evaluation review*, vol. 37, pp. 133–144, 2009.

[72] L. Jiang and J. Walrand, "A distributed CSMA algorithm for throughput and utility maximization in wireless networks," *IEEE/ACM Transactions on Networking (TON)*, vol. 18, no. 3, pp. 960–972, 2010.

[73] L. Jiang, D. Shah, J. Shin, and J. Walrand, "Distributed random access algorithm: scheduling and congestion control," *IEEE Transactions on Information Theory*, vol. 56, no. 12, pp. 6182–6207, 2010.

[74] D. Shah, J. Shin, *et al.*, "Randomized scheduling algorithm for queueing networks," *The Annals of Applied Probability*, vol. 22, no. 1, pp. 128–171, 2012.

[75] J. Ghaderi and R. Srikant, "On the design of efficient CSMA algorithms for wireless networks," in *Proc. of IEEE CDC*, pp. 954–959, IEEE, 2010.

[76] J. Ni, B. Tan, and R. Srikant, "Q-CSMA: Queue-length-based CSMA/CA algorithms for achieving maximum throughput and low delay in wireless networks," *IEEE/ACM Transactions on Networking (TON)*, vol. 20, no. 3, pp. 825–836, 2012.

[77] D. Shah, J. Shin, and P. Tetali, "Medium access using queues," in *Foundations of Computer Science (FOCS), 2011 IEEE 52nd Annual Symposium on*, pp. 698–707, IEEE, 2011.

[78] M. Lotfinezhad and P. Marbach, "Throughput-optimal random access with order-optimal delay," in *Proc. of IEEE INFOCOM*, pp. 2867–2875, 2011.

[79] E. Paolini, C. Stefanovic, G. Liva, and P. Popovski, "Coded random access: applying codes on graphs to design random access protocols," *IEEE Communications Magazine*, vol. 53, no. 6, pp. 144–150, 2015.

[80] B. Li and A. Eryilmaz, "Optimal distributed scheduling under time-varying conditions: A fast-csma algorithm with applications," *IEEE Transactions on Wireless Communications*, vol. 12, no. 7, pp. 3278–3288, 2013.

[81] N. Lu, B. Li, R. Srikant, and L. Ying, "Optimal distributed scheduling of real-time traffic with hard deadlines," in *Proc. of IEEE Conference on Decision and Control (CDC)*, pp. 4408–4413, 2016.

[82] G. Bianchi, "Performance analysis of the IEEE 802.11 distributed coordination function," *IEEE Journal on selected areas in communications*, vol. 18, no. 3, pp. 535–547, 2000.

[83] P.-K. Huang and X. Lin, "Improving the delay performance of CSMA algorithms: A virtual multi-channel approach," in *Proc. of IEEE INFOCOM*, pp. 2598–2606, 2013.

[84] C.-H. Lee, S.-Y. Yun, Y. Yi, *et al.*, "From Glauber dynamics to Metropolis algorithm: Smaller delay in optimal CSMA," in *Proc. of IEEE ISIT*, pp. 2681–2685, IEEE, 2012.

[85] K.-K. Lam, C.-K. Chau, M. Chen, and S.-C. Liew, "Mixing time and temporal starvation of general CSMA networks with multiple frequency agility," in *Proc. of IEEE ISIT*, pp. 2676–2680, IEEE, 2012.

[86] D. Shah, N. David, and J. N. Tsitsiklis, "Hardness of low delay network scheduling," *IEEE Transactions on Information Theory*, vol. 57, no. 12, pp. 7810–7817, 2011.

[87] D. Shah and J. Shin, "Delay optimal queue-based CSMA," in *ACM SIGMETRICS Performance Evaluation Review*, vol. 38, pp. 373–374, ACM, 2010.

[88] L. Jiang, M. Leconte, J. Ni, R. Srikant, and J. Walrand, "Fast mixing of parallel Glauber dynamics and low-delay CSMA scheduling," *IEEE Transactions on Information Theory*, vol. 58, no. 10, pp. 6541–6555, 2012.

[89] V. G. Subramanian and M. Alanyali, "Delay performance of CSMA in networks with bounded degree conflict graphs," in *Proc. of IEEE ISIT*, pp. 2373–2377, IEEE, 2011.

[90] K. Xu, M. Gerla, and S. Bae, "How effective is the IEEE 802.11 RTS/CTS handshake in ad hoc networks," in *Proc. of GLOBECOM*, vol. 1, pp. 72–76, 2002.

[91] F. J. Beutler and K. W. Ross, "Optimal policies for controlled Markov chains with a constraint," *Journal of mathematical analysis and applications*, vol. 112, no. 1, pp. 236–252, 1985.

[92] A. Eryilmaz, R. Srikant, and J. R. Perkins, "Stable scheduling policies for fading wireless channels," *IEEE/ACM Transactions on Networking (TON)*, vol. 13, no. 2, pp. 411–424, 2005.

[93] P.-C. Hsieh and I.-H. Hou, "A decentralized medium access protocol for real-time wireless ad hoc networks with unreliable transmissions," tech. rep., February 2018.

[94] E. Magistretti, K. K. Chintalapudi, B. Radunovic, and R. Ramjee, "Wifi-nano: reclaiming wifi efficiency through 800 ns slots," in *Proc. of International Conference on Mobile computing and networking*, pp. 37–48, ACM, 2011.

[95] IEEE 802 Standard Working Group, "Wireless LAN medium access control (MAC) and physical layer (PHY) specifications: high-speed physical layer in the 5GHz Band," 1999.

[96] H. Chen and D. D. Yao, *Fundamentals of queueing networks: Performance, asymptotics, and optimization*, vol. 46. Springer Science & Business Media, 2001.

[97] D. N. C. Tse, R. G. Gallager, and J. N. Tsitsiklis, "Statistical multiplexing of multiple time-scale Markov streams," *IEEE Journal on Selected Areas in Communications*, vol. 13, no. 6, pp. 1028–1038, 1995.

[98] G. G. Yin and Q. Zhang, *Discrete-time Markov chains: two-time-scale methods and applications*, vol. 55. Springer, 2006.

[99] `https://github.com/pinghsieh/ns3-sim`.

[100] I. F. Akyildiz, T. Melodia, and K. R. Chowdhury, "A survey on wireless multimedia sensor networks," *Computer networks*, vol. 51, no. 4, pp. 921–960, 2007.

[101] A. Frotzscher, U. Wetzker, M. Bauer, M. Rentschler, M. Beyer, S. Elspass, and H. Klessig, "Requirements and current solutions of wireless communication in industrial automation," in *Proc. of IEEE International Conference on Communications Workshops (ICC)*, pp. 67–72, 2014.

[102] J. Åkerberg, M. Gidlund, and M. Björkman, "Future research challenges in wireless sensor and actuator networks targeting industrial automation," in *Proc. of IEEE International Conference on Industrial Informatics (INDIN)*, pp. 410–415, 2011.

[103] S. Yau, P.-C. Hsieh, R. Bhattacharyya, K. Bhargav K. R., S. Shakkottai, I.-H. Hou, and P. R. Kumar, "Demo: PULS: Processor-Supported Ultra-Low Latency Scheduling," in *Proc. of ACM MobiHoc*, 2018.

[104] "NSF Workshop on Ultra-Low Latency Wireless Networks," November 2016.

[105] ITU-T, "The Tactile Internet," August 2014.

[106] H. Holma and A. Toskala, *LTE for UMTS: Evolution to LTE-advanced*. John Wiley & Sons, 2011.

[107] M. Laner, P. Svoboda, P. Romirer-Maierhofer, N. Nikaein, F. Ricciato, and M. Rupp, "A comparison between one-way delays in operating HSPA and LTE networks," in *Modeling and Optimization in Mobile, Ad Hoc and Wireless Networks (WiOpt), 2012 10th International Symposium on*, pp. 286–292, IEEE, 2012.

[108] K. Sui, M. Zhou, D. Liu, M. Ma, D. Pei, Y. Zhao, Z. Li, and T. Moscibroda, "Characterizing and improving WiFi latency in large-scale operational net-

works," in *Proceedings of the 14th Annual International Conference on Mobile Systems, Applications, and Services*, pp. 347–360, ACM, 2016.

[109] I. H. Hou, V. Borkar, and P. R. Kumar, "A Theory of QoS for Wireless," in *Proc. of IEEE INFOCOM*, pp. 486–494, April 2009.

[110] I.-H. Hou and P. Kumar, "Utility maximization for delay constrained qos in wireless," in *Proc. of IEEE INFOCOM*, pp. 1–9, 2010.

[111] J. J. Jaramillo and R. Srikant, "Optimal Scheduling for Fair Resource Allocation in Ad Hoc Networks With Elastic and Inelastic Traffic," *IEEE/ACM Transactions on Networking*, vol. 4, no. 19, pp. 1125–1136, 2011.

[112] G. Nychis, T. Hottelier, Z. Yang, S. Seshan, and P. Steenkiste, "Enabling mac protocol implementations on software-defined radios.," in *NSDI*, vol. 9, pp. 91–105, 2009.

[113] J. Ansari, X. Zhang, A. Achtzehn, M. Petrova, and P. Mahonen, "Decomposable mac framework for highly flexible and adaptable mac realizations," in *New Frontiers in Dynamic Spectrum, 2010 IEEE Symposium on*, pp. 1–2, IEEE, 2010.

[114] A. Warrier, S. Janakiraman, S. Ha, and I. Rhee, "DiffQ: Practical differential backlog congestion control for wireless networks," in *Proc. of IEEE INFOCOM*, pp. 262–270, 2009.

[115] R. Laufer, T. Salonidis, H. Lundgren, and P. Le Guyadec, "XPRESS: A cross-layer backpressure architecture for wireless multi-hop networks," in *Proceedings of the 17th annual international conference on Mobile computing and networking*, pp. 49–60, ACM, 2011.

[116] J. Ryu, V. Bhargava, N. Paine, and S. Shakkottai, "Back-pressure routing and rate control for ICNs," in *Proceedings of the sixteenth annual international conference on Mobile computing and networking*, pp. 365–376, ACM, 2010.

[117] J. Lee, H. Lee, Y. Yi, S. Chong, E. W. Knightly, and M. Chiang, "Making 802.11 DCF near-optimal: Design, implementation, and evaluation," *IEEE/ACM Transactions on Networking*, vol. 24, no. 3, pp. 1745–1758, 2016.

[118] Y.-H. Wei, Q. Leng, S. Han, A. K. Mok, W. Zhang, and M. Tomizuka, "RT-WiFi: Real-time high-speed communication protocol for wireless cyber-physical control applications," in *Proc. of Real-Time Systems Symposium (RTSS)*, pp. 140–149, IEEE, 2013.

[119] O. N. Yilmaz, Y.-P. E. Wang, N. A. Johansson, N. Brahmi, S. A. Ashraf, and J. Sachs, "Analysis of ultra-reliable and low-latency 5G communication for a factory automation use case," in *Communication Workshop (ICCW), 2015 IEEE International Conference on*, pp. 1190–1195, IEEE, 2015.

[120] S. Yoshioka, Y. Inoue, S. Suyama, Y. Kishiyama, Y. Okumura, J. Kepler, and M. Cudak, "Field experimental evaluation of beamtracking and latency performance for 5G mmWave radio access in outdoor mobile environment," in *Personal, Indoor, and Mobile Radio Communications (PIMRC), 2016 IEEE 27th Annual International Symposium on*, pp. 1–6, IEEE, 2016.

[121] J. Pilz, M. Mehlhose, T. Wirth, D. Wieruch, B. Holfeld, and T. Haustein, "A Tactile Internet demonstration: 1ms ultra low delay for wireless communications towards 5G," in *Proc. of IEEE INFOCOM Workshops*, pp. 862–863, 2016.

[122] "Labview communications system design suite." `http://www.ni.com/labview-communications/`.

[123] "Labview communications 802.11 application framework." http://www.ni.com/white-paper/53279/en/.

[124] I. Tinnirello, G. Bianchi, P. Gallo, D. Garlisi, F. Giuliano, and F. Gringoli, "Wireless MAC processors: Programming MAC protocols on commodity hardware," in *Proc. of IEEE INFOCOM*, pp. 1269–1277, 2012.

[125] P. Giaccone, E. Leonardi, and D. Shah, "Throughput region of finite-buffered networks," *IEEE Transactions on Parallel and Distributed Systems*, vol. 18, no. 2, pp. 251–263, 2007.

[126] C. D. Meyer, *Matrix analysis and applied linear algebra*, vol. 2. Siam, 2000.

[127] L. Hogben, *Handbook of linear algebra*. CRC Press, 2006.

[128] M. Bramson, *Stability of queueing networks*. Springer, 2008.

APPENDIX A

PROOFS OF CHAPTER 2

A.1 Proof of Lemma 1

Proof. Given $\Delta L(t_k) = 2\mathbf{Q}(t_k)^\intercal \Delta\mathbf{Q}(t_k) + \Delta\mathbf{Q}(t_k)^\intercal \Delta\mathbf{Q}(t_k)$, we provide an upper bound for each term separately. First, we derive an upper bound of $\mathbb{E}\big[\Delta\mathbf{Q}(t_k)^\intercal \Delta\mathbf{Q}(t_k)\big|\mathbf{Q}(t_k)\big]$. Define $V_{\max} := \max\{A_{\max}, S_{\max}\}$. If a link i is an entry link and $j \in \mathcal{D}(i)$, then

$$|\Delta Q_{i,j}(t_k)| \leq T_k \max\{A_{\max}, S_{\max}\} = T_k V_{\max}. \tag{A.1}$$

Otherwise, if $i \in \mathcal{L}_{\text{int}}$ and $j \in \mathcal{D}(i)$, then we know

$$|\Delta Q_{i,j}(t_k)| \leq U_{\max} T_k S_{\max}. \tag{A.2}$$

Hence, we have

$$\mathbb{E}\big[\Delta\mathbf{Q}(t_k)^\intercal \Delta\mathbf{Q}(t_k) \,\big|\, \mathbf{Q}(t_k)\big] \leq |\mathcal{M}|\, U_{\max}^2 V_{\max}^2 T_k^2. \tag{A.3}$$

Next, we provide an upper bound on $\mathbb{E}\big[\mathbf{Q}(t_k)^\intercal \Delta\mathbf{Q}(t_k) \,\big|\, \mathbf{Q}(t_k)\big]$.

$$\mathbf{Q}(t_k)^\intercal \Delta\mathbf{Q}(t_k) \tag{A.4}$$

$$= \sum_{t=t_k}^{t_{k+1}-1} \sum_{(i,j)\in\mathcal{M}} \Bigg[-Q_{i,j}(t)\Big(S_{i,j}(t)I_{i,j}(t)X_{i,j}(t) \wedge Q_{i,j}(t)\Big) \tag{A.5}$$

$$+ \sum_{m:(m,i)} Q_{i,j}(t)\Big(S_{m,i}(t)I_{m,i}(t)X_{m,i}(t) \wedge Q_{i,j}(t)\Big)R_{i,j}(t) \Bigg] \tag{A.6}$$

$$+ \sum_{t=t_k}^{t_{k+1}-1} \sum_{i\in\mathcal{L}_{\text{entry}}, j\in\mathcal{D}(i)} Q_{i,j}(t_k)A_{i,j}(t) \tag{A.7}$$

$$= \sum_{t=t_k}^{t_{k+1}-1} \sum_{(i,j)\in\mathcal{M}} \Bigg[\Big(S_{i,j}(t)I_{i,j}(t)X_{i,j}(t) \wedge Q_{i,j}(t)\Big) \times \tag{A.8}$$

$$\Big(-Q_{i,j}(t) + \sum_{p\in\mathcal{D}(j)} R(j,p)(t)Q(j,p)(t_k)\Big) \Bigg] \tag{A.9}$$

$$+ \sum_{t=t_k}^{t_{k+1}-1} \sum_{i\in\mathcal{L}_{\text{entry}}, j\in\mathcal{D}(i)} Q_{i,j}(t_k)A_{i,j}(t). \tag{A.10}$$

Therefore, we have

$$\mathbb{E}\Big[\mathbf{Q}(t_k)^\intercal \Delta\mathbf{Q}(t_k) \,\Big|\, \mathbf{Q}(t_k)\Big] = \tag{A.11}$$

$$- \sum_{t=t_k}^{t_{k+1}-1} \sum_{(i,j)\in\mathcal{M}} \Bigg[\mathbb{E}\Big[S_{i,j}(t)I_{i,j}(t)X_{i,j}(t) \wedge Q_{i,j}(t) \,\Big|\, \mathbf{Q}(t_k)\Big] \tag{A.12}$$

$$\times W_{i,j}(t_k) \Bigg] + T_k\Bigg(\sum_{i\in\mathcal{L}_{\text{entry}}, j\in\mathcal{D}(i)} \lambda_i^* r_{i,j} Q_{i,j}(t_k)\Bigg). \tag{A.13}$$

Moreover, we can rewrite the second term of (A.13) as

$$\sum_{i\in\mathcal{L}_{\text{entry}},j\in\mathcal{D}(i)} \lambda_i^* r_{i,j} Q_{i,j}(t_k) \tag{A.14}$$

$$= \sum_{i\in\mathcal{L}_{\text{entry}},j\in\mathcal{D}(i)} \lambda_i^* r_{i,j} Q_{i,j}(t_k) + \sum_{j\in\mathcal{L}_{\text{int}},p\in\mathcal{D}(j)} \lambda_j^* r_{j,p} Q_{j,p}(t_k) \tag{A.15}$$

$$- \sum_{j\in\mathcal{L}_{\text{int}},p\in\mathcal{D}(j)} \lambda_j^* r_{j,p} Q_{j,p}(t_k) \tag{A.16}$$

$$= \sum_{(i,j)\in\mathcal{M}} \lambda_i^* r_{i,j} Q_{i,j}(t_k) - \sum_{j\in\mathcal{L}_{\text{int}},p\in\mathcal{D}(j)} \lambda_j^* r_{j,p} Q_{j,p}(t_k) \tag{A.17}$$

$$= \sum_{(i,j)\in\mathcal{M}} \lambda_i^* r_{i,j} \Big(Q_{i,j}(t_k) - \sum_{p:(j,p)} r_{j,p} Q_{j,p}(t_k) \Big) \tag{A.18}$$

$$= \sum_{(i,j)\in\mathcal{M}} \lambda_i^* r_{i,j} W_{i,j}(t_k). \tag{A.19}$$

Therefore, we have

$$\mathbb{E}\Big[\mathbf{Q}(t_k)^\intercal \Delta \mathbf{Q}(t_k) \;\Big|\; \mathbf{Q}(t_k) \Big] \tag{A.20}$$

$$= \sum_{t=t_k}^{t_{k+1}-1} \sum_{(i,j)\in\mathcal{M}} \Bigg[W_{i,j}(t_k) \times \tag{A.21}$$

$$\Big(\lambda_i^* r_{i,j} - \mathbb{E}\Big[S_{i,j}(t) I_{i,j}(t) X_{i,j}(t) \wedge Q_{i,j}(t) \;\Big|\; \mathbf{Q}(t_k) \Big] \Big) \Bigg]. \tag{A.22}$$

We further decompose (A.22) into two parts α_1 and α_2:

$$\alpha_1 = \sum_{t=t_k}^{t_{k+1}-1} \sum_{(i,j)\in\mathcal{M}} \Bigg[W_{i,j}(t_k) \times \tag{A.23}$$

$$\mathbb{E}\Big[\lambda_i^* r_{i,j} - \mu_{i,j} I_{i,j}(t) X_{i,j}(t) \;\Big|\; \mathbf{Q}(t_k)\Big]\Bigg], \tag{A.24}$$

$$\alpha_2 = \sum_{t=t_k}^{t_{k+1}-1} \sum_{(i,j)\in\mathcal{M}} \Bigg[W_{i,j}(t_k) \times \mathbb{E}\Big[\mu_{i,j} I_{i,j}(t) X_{i,j}(t) \tag{A.25}$$

$$- \big(S_{i,j}(t) I_{i,j}(t) X_{i,j}(t) \wedge Q_{i,j}(t)\big) \Big| \mathbf{Q}(t_k)\Big]\Bigg]. \tag{A.26}$$

We start with α_2. For each movement (i,j), if $Q_{i,j}(t) \geq S_{\max}$, then we know

$$\mathbb{E}\Big[\mu_{i,j} I_{i,j}(t) X_{i,j}(t) - \big(S_{i,j}(t) I_{i,j}(t) X_{i,j}(t) \wedge Q_{i,j}(t)\big) \;\Big|\; \mathbf{Q}(t_k)\Big] = 0. \tag{A.27}$$

Otherwise, if $Q_{i,j}(t) < S_{\max}$, we have

$$W_{i,j}(t_k) \cdot \mathbb{E}\Big[\mu_{i,j} I_{i,j}(t) X_{i,j}(t) - \tag{A.28}$$

$$\big(S_{i,j}(t) I_{i,j}(t) X_{i,j}(t) \wedge Q_{i,j}(t) \Big| \mathbf{Q}(t_k)\big)\Big] \leq \mu_{i,j} S_{\max} \tag{A.29}$$

since we know $W_{i,j}(t_k) \leq Q_{i,j}(t_k)$ by definition. Therefore, we have the following upper bound of α_2:

$$\alpha_2 \leq \sum_{t=t_k}^{t_{k+1}-1} \sum_{(i,j)\in\mathcal{M}} \mu_{i,j} S_{\max} \leq \Bigg(\sum_{(i,j)\in\mathcal{M}} \mu_{i,j} S_{\max} \Bigg) T_k. \tag{A.30}$$

Note that (A.106) is an upper bound of α_2 regardless of the scheduling policy. Next, we consider α_1. Suppose $\boldsymbol{\lambda}$ is feasible, then by definition there must exist $\epsilon > 0$ and

$\mathbf{\Sigma} = (\Sigma_{i,j})$ in the convex hull of $\mathcal{I}_v$ for each individual intersection $v \in \mathcal{V}_C$ such that

$$\mu_{i,j}\Sigma_{i,j} > \lambda_i^* r_{i,j} + \epsilon, \quad \forall (i,j) \in \mathcal{M}_v \,. \tag{A.31}$$

Moreover, we construct another vector $\mathbf{\Sigma}^*(t_k) = (\Sigma_{i,j}^*(t_k))$ as

$$\Sigma_{i,j}^*(t_k) = \begin{cases} \frac{\lambda_i^* r_{i,j} + \epsilon}{\mu_{i,j}}, & \text{if } W_{i,j}(t_k) > 0 \\ 0, & \text{otherwise} \end{cases} \tag{A.32}$$

Note that while $\mathbf{\Sigma}$ is a fixed vector across time, we choose $\mathbf{\Sigma}^*(t_k)$ depending on the pressure $W_{i,j}(t_k)$. It is easy to verify that $\mathbf{\Sigma}^*(t_k)$ is also in the convex hull of $\mathcal{I}_v$. For any max-pressure-at-switch-over policy, we must have that at time t_k

$$\sum_{(i,j)\in\mathcal{M}_v} I_{i,j}(t_k)\mu_{i,j}W_{i,j}(t_k) \geq \sum_{(i,j)\in\mathcal{M}_v} \Sigma_{i,j}^*(t_k)\mu_{i,j}W_{i,j}(t_k). \tag{A.33}$$

Although the scheduling decision at $t_{k,l}^v$ with $l \geq 1$ cannot be determined purely by the information about $\mathbf{Q}(t_k)$, we still know that the pressure of the scheduled phase remains relatively large compared to the pressure of the scheduled phase at time t_k,

i.e.

$$\sum_{(i,j)\in\mathcal{M}_v} I_{i,j}(t^v_{k,l})\mu_{i,j}W_{i,j}(t_k) \tag{A.34}$$

$$\geq \sum_{(i,j)\in\mathcal{M}_v} I_{i,j}(t^v_{k,l})\mu_{i,j}W_{i,j}(t^v_{k,l}) \tag{A.35}$$

$$- \sum_{(i,j)\in\mathcal{M}_v} (U_{\max}+1)V_{\max}I_{i,j}(t^v_{k,l})\mu_{i,j}(t^v_{k,l}-t_k) \tag{A.36}$$

$$\geq \sum_{(i,j)\in\mathcal{M}_v} \Sigma^*_{i,j}(t_k)\mu_{i,j}W_{i,j}(t^v_{k,l}) \tag{A.37}$$

$$- \sum_{(i,j)\in\mathcal{M}_v} (U_{\max}+1)V_{\max}I_{i,j}(t^v_{k,l})\mu_{i,j}(t^v_{k,l}-t_k) \tag{A.38}$$

$$\geq \left(\sum_{(i,j)\in\mathcal{M}_v} \Sigma^*_{i,j}(t_k)\mu_{i,j}W_{i,j}(t_k)\right) - C^v_0(t^v_{k,l}-t_k), \tag{A.39}$$

where $C^v_0 = (U_{\max}+1)V_{\max}\left(\sum_{(i,j)\in\mathcal{M}_v}\mu_{i,j}\right)$ is a positive constant. (A.35) and (A.36) hold since $|W_{i,j}(t+1)-W_{i,j}(t)| \leq (U_{\max}+1)V_{\max}$ for any (i,j) and any t. Note that α_1 is a sum over all movements $(i,j)\in\mathcal{M}$. Define

$$F_v(t_k) := \sum_{t=t_k}^{t_{k+1}-1} \sum_{(i,j)\in\mathcal{M}_v} \Bigg(W_{i,j}(t_k)\times \tag{A.40}$$

$$\mathbb{E}\left[\lambda^*_i r_{i,j} - \mu_{i,j}I_{i,j}(t)X_{i,j}(t)\Big|\mathbf{Q}(t_k)\right]\Bigg). \tag{A.41}$$

Then, $\alpha_1 = \sum_{v\in\mathcal{V}} F_v(t_k)$. For an intersection v under any max-pressure-at-switch-over policy, by (A.34)-(A.39) we have

$$F_v(t_k) \le T_S M_k^v \Big(\sum_{(i,j)\in\mathcal{M}_v} \lambda_i^* r_{i,j} W_{i,j}(t_k) \Big) \tag{A.42}$$

$$+ \Big(T_k - M_k^v T_S \Big) C_0^v T_k \tag{A.43}$$

$$+ \Big(T_k - M_k^v T_S \Big) \Bigg[-\epsilon \sum_{(i,j)\in\mathcal{M}_v} \big(W_{i,j}(t_k) \big)^+ \tag{A.44}$$

$$- \sum_{(i,j)\in\mathcal{M}_v} \lambda_i^* r_{i,j} \big(W_{i,j}(t_k) \big)^- \Bigg]. \tag{A.45}$$

Since $\lambda_i^* r_{i,j} < \mu_{i,j} \le S_{\max}$, then for sufficiently small ϵ we have

$$F_v(t_k) \le T_S M_k^v S_{\max} \Big(\sum_{(i,j)\in\mathcal{M}_v} W_{i,j}(t_k) \Big) - \tag{A.46}$$

$$\epsilon \Big(T_k - M_k^v T_S \Big) \Big(\sum_{(i,j)\in\mathcal{M}_v} W_{i,j}(t_k)^+ \Big) + C_0^v T_k^2. \tag{A.47}$$

On the other hand, for an intersection v under the fixed-time control policy with cycle length D_v, we have

$$
\begin{aligned}
& F_v(t_k) && \text{(A.48)}\\
& \leq \sum_{(i,j)\in M_v} W_{i,j}(t_k)^+\Big(T_k\lambda_i^* r_{i,j} - \mu_{i,j}\Sigma_{i,j}^*\xi_v(T_k - D_v)\Big) && \text{(A.49)}\\
& \quad - \sum_{(i,j)\in M_v} W_{i,j}(t_k)^-\Big(T_k\lambda_i^* r_{i,j} - \mu_{i,j}\Sigma_{i,j}^*\xi_v(T_k + D_v)\Big) && \text{(A.50)}\\
& = -\epsilon T_k\bigg(\sum_{(i,j)\in\mathcal{M}_v} W_{i,j}(t_k)^+\bigg) - T_k\bigg(\sum_{(i,j)\in\mathcal{M}_v}\lambda_i^* r_{i,j}W_{i,j}(t_k)^-\bigg) && \text{(A.51)}\\
& \quad + \sum_{(i,j)\in\mathcal{M}_v}\mu_{i,j}\Sigma_{i,j}^*\xi_v D_v W_{i,j}(t_k)^+ && \text{(A.52)}\\
& \leq -\epsilon T_k\bigg(\sum_{(i,j)\in\mathcal{M}_v} W_{i,j}(t_k)^+\bigg) + \sum_{(i,j)\in\mathcal{M}_v}\mu_{i,j}\Sigma_{i,j}^*\xi_v D_v W_{i,j}(t_k)^+ && \text{(A.53)}
\end{aligned}
$$

where (A.53) holds for sufficiently small ϵ. In summary, by the results in (A.3), (A.106), (A.47), (A.53) as well as the fact that $W_{i,j}(t)^+ \leq Q_{i,j}(t)$, we conclude that

$$
\begin{aligned}
& \mathbb{E}\big[\Delta L(t_k) \,\big|\, \mathbf{Q}(t_k)\big] \leq -2\epsilon T_k \sum_{(i,j)\in\mathcal{M}} W_{i,j}(t_k)^+ && \text{(A.54)}\\
& + C_1\sum_{v\in\mathcal{V}_C} M_k^v\bigg(\sum_{(i,j)\in\mathcal{M}_v} W_{i,j}(t_k)^+\bigg) && \text{(A.55)}\\
& + C_2\sum_{v\in\mathcal{V}_F}\sum_{(i,j)\in\mathcal{M}_v} W_{i,j}(t_k)^+ + C_3T_k^2 + C_4T_k, && \text{(A.56)}
\end{aligned}
$$

where

$$C_1 = 2T_S\big(\epsilon + S_{\max}\big), \tag{A.57}$$

$$C_2 = 2S_{\max}\big(\max_{v\in\mathcal{V}_F} D_v\big), \tag{A.58}$$

$$C_3 = |\mathcal{M}|U_{\max}^2 V_{\max}^2 + 2\Big(\sum_{v\in\mathcal{V}_C} C_0^v\Big) \tag{A.59}$$

$$C_4 = 2\Big(\sum_{(i,j)\in\mathcal{M}} \mu_{i,j} S_{\max}\Big). \tag{A.60}$$

The proof is complete. □

A.2 Proof of Lemma 3

Proof. We briefly summarize the results of Perron-Frobenius Theorem for non-negative matrices. First, we consider the connection between a matrix and the induced directed graph.

Definition 12. *Let $\mathbf{P} = (p_{ij})$ be an $n \times n$ matrix. The graph $G(\mathbf{P})$ is defined as the directed graph on n nodes $\{N_1, ..., N_n\}$ where there is a directed edge from N_i to N_j if and only if $p_{i,j} \neq 0$, for all $i, j = 1, ..., n$.*

Definition 13. *Let $\mathbf{P} = (p_{ij})$ be an $n \times n$ matrix. $\mathbf{P}$ is said to be irreducible if and only if the corresponding directed graph $G(\mathbf{P})$ is strongly connected.*

Lemma 14. *(Perron-Frobenius Theorem [126]) Let $\mathbf{P}$ be an $n \times n$ irreducible non-negative matrix. There exists a unique strictly positive eigenvector $\mathbf{x} = (x_1, ..., x_n)$ with $\sum_{i=1}^n x_i = 1$ and the corresponding eigenvalue $\lambda_{pf} > 0$.*

Next, we consider the substochastic properties of a matrix.

Definition 14. *A non-negative square matrix $\mathbf{P}$ is said to be substochastic if every*

row sum of $\mathbf{P}$ *is less than or equal to 1 and at least one row sum is strictly less than 1.*

Lemma 15. *(Section 9.4 in [127]) Let* $\mathbf{P}$ *be an* $n \times n$ *substochastic matrix and* $\mathbf{1}$ *be an* $n \times n$ *identity matrix. Then, the maximum eigenvalue of* $\mathbf{P}$ *is less than 1 if and only if* $(\mathbf{1} - \mathbf{P})$ *is invertible.*

Now, we are ready to prove Lemma 3. Given a multi-hop system $\mathcal{G} = (\mathcal{V}, \mathcal{L}, \mathcal{M})$, we construct another system $\mathcal{G}' = (\mathcal{V}', \mathcal{L}', \mathcal{M}')$ by adding both dummy links $\mathcal{L}_{\text{dummy}}$ and dummy movements $\mathcal{M}_{\text{dummy}}$ to the original system $\mathcal{G}$. Specifically, for every node v other than the source node v_{s} and destination node v_{d}, we add an extra link from the v_{d} to v. For every dummy link i from v_{d} to v, we further add a new movement from link i to link j for every $j \in \mathcal{L}$ with an associated dummy queue $Q_{i,j}$. Therefore, for every dummy link i from v_{d} to v, we have $\mathcal{D}(i) = \mathcal{L}$. Moreover, for every dummy link $i \in \mathcal{L}_{\text{dummy}}$, the routing probability $r_{i,j}$ from link i to a downstream link j is set to be $\frac{1}{2|\mathcal{L}|}$, for every $j \in \mathcal{D}(i)$. Therefore, $\sum_{j:j\in\mathcal{D}(i), i\in\mathcal{L}_{\text{dummy}}} = \frac{1}{2} < 1$. Therefore, the constructed graph $\mathcal{G}'$ is strongly connected. Let $\widetilde{\mathbf{Q}} = (Q_{i,j})_{(i,j)\in\mathcal{M}'}$ and $\widetilde{\mathbf{W}} = (W_{i,j})_{(i,j)\in\mathcal{M}'}$ be the corresponding queue length vector and pressure vector of the constructed system $\mathcal{G}'$, respectively. Then, we have

$$\widetilde{\mathbf{W}} = (\mathbf{1} - \widetilde{\mathbf{R}})\widetilde{\mathbf{Q}}, \tag{A.61}$$

where $\mathbf{1}$ is an $|\mathcal{L}'| \times |\mathcal{L}'|$ identity matrix and $\widetilde{\mathbf{R}}$ is the corresponding routing matrix with entries $\{r_{i,j}\}$ for all movements $(i,j) \in \mathcal{M}'$. It is easy to verify that $\widetilde{\mathbf{R}}$ is a substochastic matrix and $(\mathbf{1} - \widetilde{\mathbf{R}})$ is invertible by using elementary linear algebra. Note that for any dummy movement $(i,j) \in \mathcal{M}' \setminus \mathcal{M}$, we are allowed to freely assign values to $Q_{i,j}$ in any time slot and here we assign $Q_{i,j} = 0$. Therefore, the

corresponding $W_{i,j}$ is always non-positive, for any $(i,j) \in \mathcal{M}' \setminus \mathcal{M}$.

Since $\mathcal{G}'$ is strongly connected, then the routing matrix $\widetilde{\mathbf{R}}$ is irreducible. By Lemma 14, there exists a unique strictly positive eigenvector $\mathbf{x} = (x_{i,j})_{(i,j)\in\mathcal{M}'}$ of $\widetilde{\mathbf{R}}$ with $\sum_{(i,j)\in\mathcal{M}'} x_{i,j} = 1$ and the corresponding eigenvalue $\lambda_{\text{pf}} > 0$. Hence,

$$\mathbf{x}^\intercal \widetilde{\mathbf{R}} = \lambda_{\text{pf}} \mathbf{x}^\intercal. \tag{A.62}$$

Moreover, since $\widetilde{\mathbf{R}}$ is substochastic and $(\mathbf{1} - \widetilde{\mathbf{R}})$ is invertible, by Lemma 15 we also have $\lambda_{pf} < 1$. From (A.61) and (A.62), we have

$$\mathbf{x}^\intercal \widetilde{\mathbf{W}} = \mathbf{x}^\intercal (\mathbf{1} - \widetilde{\mathbf{R}}) \widetilde{\mathbf{Q}} = (1 - \lambda_{\text{pf}}) \mathbf{x}^\intercal \widetilde{\mathbf{Q}} \tag{A.63}$$

Therefore, $\mathbf{x}^\intercal \left(\widetilde{\mathbf{W}} - (1 - \lambda_{\text{pf}}) \widetilde{\mathbf{Q}} \right) = 0$. In other words, $\widetilde{\mathbf{W}}$ is in the perpendicular complement of the vector space spanned by $\mathbf{x}$. Hence, we can write $\widetilde{\mathbf{W}}$ as

$$\widetilde{\mathbf{W}} = (1 - \lambda_{\text{pf}}) \widetilde{\mathbf{Q}} + \mathbf{y}, \tag{A.64}$$

where $\mathbf{y} = (y_{i,j})_{(i,j)\in\mathcal{M}'}$ is a vector orthogonal to the vector $\mathbf{x}$. Next, we consider two cases of $\mathbf{y}$:

Case 1: $\mathbf{y} = \mathbf{0}$

Then, we directly have $\widetilde{\mathbf{W}} = (1 - \lambda_{\text{pf}}) \widetilde{\mathbf{Q}}$. Since $\widetilde{\mathbf{Q}}$ is non-negative, then $\widetilde{\mathbf{W}}$ is also non-negative. Moreover, since we assign $Q_{i,j} = 0$ for every $(i,j) \in \mathcal{M}' \setminus \mathcal{M}$, then $W_{i,j} = 0$, $\forall (i,j) \in \mathcal{M}' \setminus \mathcal{M}$. Hence, it is easy to verify that

$$\sum_{(i,j)\in\mathcal{M}} {W_{i,j}}^+ = (1 - \lambda_{\text{pf}}) \sum_{(i,j)\in\mathcal{M}} Q_{i,j}. \tag{A.65}$$

Case 2: $\mathbf{y} \neq \mathbf{0}$

Since $\mathbf{x}$ is strictly positive and $\mathbf{x}^\intercal \mathbf{y} = 0$, then $\mathbf{y}$ cannot be non-positive. Let (i_q^*, j_q^*) be the movement with the largest queue length in $\widetilde{\mathbf{Q}}$ and (i_y^*, j_y^*) be the movement with largest entry in $\mathbf{y}$. Fix a small $\delta_1 > 0$.

- If $W_{i_q^*,j_q^*} \geq \delta_1 Q_{i_q^*,j_q^*}$, then we have

$$\sum_{(i,j)\in\mathcal{M}} {W_{i,j}}^+ \geq \delta_1 Q_{i_q^*,j_q^*} \geq \frac{\delta_1}{|\mathcal{M}|} \sum_{(i,j)\in\mathcal{M}} Q_{i,j}. \tag{A.66}$$

- If $W_{i_q^*,j_q^*} < \delta_1 Q_{i_q^*,j_q^*}$, then we know

$$y_{i_q^*,j_q^*} = W_{i_q^*,j_q^*} - (1-\lambda_{\text{pf}}) Q_{i_q^*,j_q^*} < -(1-\lambda_{\text{pf}}-\delta_1) Q_{i_q^*,j_q^*}. \tag{A.67}$$

  Since $\mathbf{x}^\intercal \mathbf{y} = \sum_{(i,j)} x_{i,j} y_{i,j} = 0$, then

$$x_{i_y^*,j_y^*} y_{i_y^*,j_y^*} \geq -\frac{y_{i_q^*,j_q^*}}{|\mathcal{M}'|} > \frac{1}{|\mathcal{M}'|}(1-\lambda_{\text{pf}}-\delta_1) Q_{i_q^*,j_q^*}. \tag{A.68}$$

  Therefore, it is easy to verify that

$$\sum_{(i,j)\in\mathcal{M}} {W_{i,j}}^+ \geq y_{i_y^*,j_y^*} \geq \frac{1-\lambda_{\text{pf}}-\delta_1}{x_{i_y^*,j_y^*}|\mathcal{M}'||\mathcal{M}|} \sum_{(i,j)\in\mathcal{M}} Q_{i,j}. \tag{A.69}$$

In summary, by (A.65)-(A.69), there always exists a constant $\delta > 0$ such that $\sum_{(i,j)\in\mathcal{M}} {W_{i,j}}^+ \geq \delta\Big(\sum_{(i,j)\in\mathcal{M}} Q_{i,j}\Big)$. □

A.3 Proof of Lemma 4

Proof. Consider the switch-over condition (2.11) in the l-th frame of the k-th superframe. For any $t \in [t_{k,l}^v, t_{k,l+1}^v]$, the right-hand side of (2.11) is upper bounded

as

$$\Big(\sum_{(i,j)\in\mathcal{M}_v}\mu_{i,j}I^*_{i,j}(t)W_{i,j}(t)\Big)^+ \tag{A.70}$$

$$\leq\Big(\sum_{(i,j)\in\mathcal{M}_v}\mu_{i,j}I^*_{i,j}(t)W_{i,j}(t^v_{k,l})\Big)^+ + |\mathcal{M}_v|\,(U_{\max}+1)V_{\max}T^v_{k,l}, \tag{A.71}$$

where $T^v_{k,l} := t^v_{k,l+1} - t^v_{k,l}$, for all k and l. Similarly, we have a lower bound for the left-hand side of (2.10):

$$\Big(1+B_v(t^v_{k,l})\Big)\Big(\sum_{(i,j)\in\mathcal{M}_v}\mu_{i,j}I^*_{i,j}(t^v_{k,l})W_{i,j}(t)\Big)^+ \tag{A.72}$$

$$\geq\Big(1+B_v(t^v_{k,l})\Big)\Big(\sum_{(i,j)\in\mathcal{M}_v}\mu_{i,j}I^*_{i,j}(t^v_{k,l})W_{i,j}(t^v_{k,l})\Big)^+ \tag{A.73}$$

$$-\Big(1+B_v(t^v_{k,l})\Big)\,|\mathcal{M}_v|\,(U_{\max}+1)V_{\max}T^v_{k,l}. \tag{A.74}$$

Therefore, since $B_v(t^v_{k,l}) \leq \zeta T_S$, we have

$$(2+\zeta T_S)\,|\mathcal{M}_v|\,(U_{\max}+1)V_{\max}T^v_{k,l} \tag{A.75}$$

$$\geq B_v(t^v_{k,l})\Big(\sum_{(i,j)\in\mathcal{M}_v}\mu_{i,j}I^*_{i,j}(t^v_{k,l})W_{i,j}(t^v_{k,l})\Big)^+ \tag{A.76}$$

$$\geq B_v(t^v_{k,l})\cdot\max_{(i,j)\in\mathcal{M}_v}\mu_{i,j}W_{i,j}(t^v_{k,l})^+ \tag{A.77}$$

$$\geq B_v(t^v_{k,l})\frac{\mu_{\min}}{|\mathcal{M}_v|}\Big(\sum_{(i,j)\in\mathcal{M}_v}W_{i,j}(t^v_{k,l})^+\Big), \tag{A.78}$$

where $\mu_{\min} := \min_{(i,j)\in\mathcal{M}}\mu_{i,j} > 0$. Hence, we conclude that

$$T^v_{k,l} \geq C_5 B_v(t^v_{k,l})\Big(\sum_{(i,j)\in\mathcal{M}_v}W_{i,j}(t^v_{k,l})^+\Big), \tag{A.79}$$

where $C_5 = \mu_{\max}\Big((2+\zeta T_S)\left|\mathcal{M}_v\right|^2 (U_{\max}+1)V_{\max}\Big)^{-1}$. □

A.4 Proof of Lemma 5

Proof. Consider the following two cases:

Case 1: $\sum_{(i,j)\in\mathcal{M}_v} W_{i,j}(t)^+ < 1$ for some $t \in [t_k, t_{k+1})$

Since $M_k^v \le T_k$ and $|W_{i,j}(t+1) - W_{i,j}(t)| \le (U_{\max}+1)V_{\max}$ for any (i,j) and any t, then we have

$$\begin{aligned} M_k^v\Big(\sum_{(i,j)\in\mathcal{M}_v} W_{i,j}(t_k)^+\Big) &< T_k\left|\mathcal{M}_v\right|\Big(1+(U_{\max}+1)V_{\max}T_k\Big) \\ &\le C_6\Big(\sum_{(i,j)\in\mathcal{M}} Q_{i,j}(t_k)\Big)^{2\beta}, \end{aligned}$$

where $C_6 = \left|\mathcal{M}_v\right|\Big(1+(U_{\max}+1)V_{\max}\Big)$.

Case 2: $\sum_{(i,j)\in\mathcal{M}_v} W_{i,j}(t)^+ \ge 1$ for all $t \in [t_k, t_{k+1})$

In this case, it is easy to verify that for any $t \in [t_k, t_{k+1})$,

$$\min\left\{1, \left(\Big[\sum_{(i,j)\in\mathcal{M}_v} W_{i,j}(t)\Big]^+\right)^{-\alpha}\right\} \ge \left(\sum_{(i,j)\in\mathcal{M}_v} W_{i,j}(t)^+\right)^{-\alpha}. \tag{A.80}$$

Therefore, at each $t_{k,l}^v$ the bias function is lower bounded as

$$B_v(t_{k,l}^v) \ge \zeta T_S\Big(\sum_{(i,j)\in\mathcal{M}_v} W_{i,j}(t_{k,l}^v)^+\Big)^{-\alpha}. \tag{A.81}$$

By Lemma 4, we have

$$T_{k,l}^{v} \geq C_5 \zeta T_S \Big(\sum_{(i,j)\in\mathcal{M}_v} W_{i,j}(t_{k,l}^{v})^{+} \Big)^{1-\alpha} \tag{A.82}$$

$$\geq C_5 \zeta T_S \Big[\sum_{(i,j)\in\mathcal{M}_v} \Big(W_{i,j}(t_k)^{+} - (U_{\max}+1)V_{\max}T_k \Big)^{+} \Big]^{1-\alpha} \tag{A.83}$$

$$\geq C_5 \zeta T_S \Bigg[\frac{1}{2^{1-\alpha}} \Big(\sum_{(i,j)\in\mathcal{M}_v} W_{i,j}(t_k)^{+} \Big)^{1-\alpha} \tag{A.84}$$

$$- \Big((U_{\max}+1)V_{\max}T_k \Big)^{1-\alpha} \Bigg] \tag{A.85}$$

where the last inequality holds since $|a+b|^p \leq 2^p\big(|a|^p + |b|^p\big)$, for any $a, b \in \mathbb{R}$ and for any $p > 0$. Next, we need to discuss the following two possible scenarios:

Case 2-1:

$$\frac{1}{2^{1-\alpha}} \Big(\sum_{(i,j)\in\mathcal{M}_v} W_{i,j}(t_k)^{+} \Big)^{1-\alpha} \geq 2 \Big((U_{\max}+1)V_{\max}T_k \Big)^{1-\alpha} \tag{A.86}$$

Then, we have a lower bound on $T_{k,l}^{v}$ as

$$T_{k,l}^{v} \geq C_5 \zeta T_S \Big((U_{\max}+1)V_{\max}T_k \Big)^{1-\alpha} \tag{A.87}$$

Without loss of generality, we only need to consider the case where $C_5 \zeta T_S \Big((U_{\max}+1)V_{\max}T_k \Big)^{1-\alpha} > 1$ (Otherwise, T_k is upper bounded by a constant). Therefore, we have

$$M_k^{v} \Big(\sum_{(i,j)\in\mathcal{M}_v} W_{i,j}(t_k)^{+} \Big) \tag{A.88}$$

$$\leq \frac{T_k}{C_5 \zeta T_S \Big((U_{\max}+1)V_{\max}T_k \Big)^{1-\alpha} - 1} \cdot \Big(\sum_{(i,j)\in\mathcal{M}_v} W_{i,j}(t_k)^{+} \Big). \tag{A.89}$$

Since $\Big(\sum_{(i,j)\in\mathcal{M}_v} W_{i,j}(t_k)^+\Big) \leq \Big(\sum_{(i,j)\in\mathcal{M}} Q_{i,j}(t_k)\Big)$, there exists a constant $C_7 > 0$ such that

$$M_k^v\Big(\sum_{(i,j)\in\mathcal{M}_v} W_{i,j}(t_k)^+\Big) \leq C_7 T_k^\alpha \sum_{(i,j)\in\mathcal{M}} Q_{i,j}(t_k) \tag{A.90}$$

$$= C_7\Big(\sum_{(i,j)\in\mathcal{M}} Q_{i,j}(t_k)\Big)^{1+\alpha\beta}. \tag{A.91}$$

Case 2-2:

$$\frac{1}{2^{1-\alpha}}\Big(\sum_{(i,j)\in\mathcal{M}_v} W_{i,j}(t_k)^+\Big)^{1-\alpha} < 2\Big((U_{\max}+1)V_{\max}T_k\Big)^{1-\alpha} \tag{A.92}$$

In this case, it is easy to verify that there exists a constant $C_8 > 0$ such that

$$M_k^v\Big(\sum_{(i,j)\in\mathcal{M}_v} W_{i,j}(t_k)^+\Big) \leq C_8 T_k^2 = C_8\Big(\sum_{(i,j)\in\mathcal{M}} Q_{i,j}(t_k)\Big)^{2\beta}. \tag{A.93}$$

Therefore, the proof is complete. □

A.5 Proof of Lemma 6

Proof. We still consider the Lyapunov function $L(\mathbf{Q}(t)) = \sum_{(i,j)\in\mathcal{M}} Q_{i,j}(t)^2$. By the condition that $Q_{i,j}(t) - Q_{i,j}^\dagger(t) \in [-B, B]$, we also have $W_{i,j}(t) - W_{i,j}^\dagger(t) \in [-2B, 2B]$, for all (i, j) and all t. Similar to the proof of Lemma 1, we consider the conditional drift over one superframe:

$$\mathbb{E}\Big[\Delta L(t_k) \;\Big|\; \mathbf{Q}^\dagger(t_k)\Big] = \mathbb{E}\Big[2\mathbf{Q}(t_k)^\intercal \Delta\mathbf{Q}(t_k) + \Delta\mathbf{Q}^\intercal\Delta\mathbf{Q}(t_k) \;\Big|\; \mathbf{Q}^\dagger(t_k)\Big]. \tag{A.94}$$

First, similar to (A.3), we have

$$\mathbb{E}\big[\Delta\mathbf{Q}(t_k)^\intercal \Delta\mathbf{Q}(t_k) \,\big|\, \mathbf{Q}^\dagger(t_k)\big] \le |\mathcal{M}|\, U_{\max}^2 V_{\max}^2 T_k^2. \tag{A.95}$$

Next, we consider $\mathbb{E}\big[\mathbf{Q}(t_k)^\intercal \Delta\mathbf{Q}(t_k) \,\big|\, \mathbf{Q}^\dagger(t_k)\big]$. Following the same procedure as in (A.4)-(A.22), we have

$$\begin{aligned}
&\mathbb{E}\Big[\mathbf{Q}(t_k)^\intercal \Delta\mathbf{Q}(t_k) \mid \mathbf{Q}^\dagger(t_k)\Big] && \text{(A.96)}\\
&= \sum_{t=t_k}^{t_{k+1}-1} \sum_{(i,j)\in\mathcal{M}} \Bigg[\lambda_i^* r_{i,j}\, \mathbb{E}\Big[W_{i,j}(t_k) \mid \mathbf{Q}^\dagger(t_k)\Big] && \text{(A.97)}\\
&\quad - \mathbb{E}\Big[S_{i,j}(t) I_{i,j}(t) X_{i,j}(t) \wedge Q_{i,j}(t) \,\Big|\, \mathbf{Q}^\dagger(t_k)\Big]\Bigg] && \text{(A.98)}\\
&\le \sum_{t=t_k}^{t_{k+1}-1} \sum_{(i,j)\in\mathcal{M}} \Bigg[\lambda_i^* r_{i,j}\, \mathbb{E}\Big[W_{i,j}^\dagger(t_k) \mid \mathbf{Q}^\dagger(t_k)\Big] && \text{(A.99)}\\
&\quad - \mathbb{E}\Big[S_{i,j}(t) I_{i,j}(t) X_{i,j}(t) \wedge Q_{i,j}(t) \,\Big|\, \mathbf{Q}^\dagger(t_k)\Big]\Bigg] && \text{(A.100)}\\
&\quad + |\mathcal{M}|\, A_{\max} B T_k + |\mathcal{M}|\, S_{\max} B T_k. && \text{(A.101)}
\end{aligned}$$

Now, as in (A.23)-(A.26), we further decompose (A.99)-(A.100) into two parts:

$$\begin{aligned}
\alpha_1^\dagger = \sum_{t=t_k}^{t_{k+1}-1} \sum_{(i,j)\in\mathcal{M}} \Bigg[& W_{i,j}^\dagger(t_k) \times && \text{(A.102)}\\
& \mathbb{E}\Big[\lambda_i^* r_{i,j} - \mu_{i,j} I_{i,j}(t) X_{i,j}(t) \,\Big|\, \mathbf{Q}^\dagger(t_k)\Big]\Bigg], && \text{(A.103)}\\
\alpha_2^\dagger = \sum_{t=t_k}^{t_{k+1}-1} \sum_{(i,j)\in\mathcal{M}} \Bigg[& W_{i,j}^\dagger(t_k) \times \mathbb{E}\Big[\mu_{i,j} I_{i,j}(t) X_{i,j}(t) && \text{(A.104)}\\
& - \big(S_{i,j}(t) I_{i,j}(t) X_{i,j}(t) \wedge Q_{i,j}(t)\big) \Big| \mathbf{Q}^\dagger(t_k)\Big]\Bigg]. && \text{(A.105)}
\end{aligned}$$

By a similar argument as in (A.27)-(A.29), we know

$$\alpha_2^\dagger \leq \Bigg(\sum_{(i,j)\in\mathcal{M}} \mu_{i,j}(S_{\max} + B) \Bigg) T_k \tag{A.106}$$

since $W_{i,j}^\dagger(t_k) \leq Q_{i,j}^\dagger(t_k) \leq Q_{i,j}(t_k) + B$, for any (i,j). To calculate $\alpha_1^\dagger$, as in (A.32) we construct a vector $\mathbf{\Sigma}^{**}(t_k) = (\Sigma_{i,j}^{**}(t_k))$ as

$$\Sigma_{i,j}^{**}(t_k) = \begin{cases} \frac{\lambda_i^* r_{i,j} + \epsilon}{\mu_{i,j}}, & \text{if } W_{i,j}^\dagger(t_k) > 0 \\ 0, & \text{otherwise} \end{cases} \tag{A.107}$$

Again, by the max-pressure-at-switch-over property, we have

$$\sum_{(i,j)\in\mathcal{M}_v} I_{i,j}(t_k)\mu_{i,j}W_{i,j}^\dagger(t_k) \geq \sum_{(i,j)\in\mathcal{M}_v} \Sigma_{i,j}^{**}(t_k)\mu_{i,j}W_{i,j}^\dagger(t_k). \tag{A.108}$$

Following the same procedure as in (A.34)-(A.39), we have

$$\sum_{(i,j)\in\mathcal{M}_v} I_{i,j}(t_{k,l}^v)\mu_{i,j}W_{i,j}^\dagger(t_k) \tag{A.109}$$

$$\geq \Bigg(\sum_{(i,j)\in\mathcal{M}_v} \Sigma_{i,j}^{**}(t_k)\mu_{i,j}W_{i,j}^\dagger(t_k) \Bigg) - C_0^{\dagger,v}(t_{k,l}^v - t_k), \tag{A.110}$$

where $C_0^{\dagger,v} = (4B + (U_{\max}+1)V_{\max}) \cdot \big(\sum_{(i,j)\in\mathcal{M}_v} \mu_{i,j}\big)$. Following the same discussion as in (A.40)-(A.53) with $W_{i,j}(t_k)$ replaced by $W_{i,j}^{\dagger}(t_k)$, we have

$$\mathbb{E}\big[\Delta L(t_k) \,\big|\, \mathbf{Q}^{\dagger}(t_k)\big] \leq -2\epsilon T_k \sum_{(i,j)\in\mathcal{M}} W_{i,j}^{\dagger}(t_k)^{+} \tag{A.111}$$

$$+ C_1^{\dagger} \sum_{v\in\mathcal{V}_C} M_k^v \bigg(\sum_{(i,j)\in\mathcal{M}_v} W_{i,j}^{\dagger}(t_k)^{+} \bigg) \tag{A.112}$$

$$+ C_2^{\dagger} \sum_{v\in\mathcal{V}_F} \sum_{(i,j)\in\mathcal{M}_v} W_{i,j}^{\dagger}(t_k)^{+} + C_3^{\dagger} T_k^2 + C_4^{\dagger} T_k, \tag{A.113}$$

where $C_1^{\dagger}, C_2^{\dagger}, C_3^{\dagger}$, and $C_4^{\dagger}$ are some positive constants. □

APPENDIX B

PROOFS OF CHAPTER 3

B.1 Proof of Theorem 9

By Lemma 7, we can write down the necessary and sufficient conditions for independent ON-OFF channels as

$$1 - \prod_{n \in S}(1 - p_n) \geq \frac{1}{r^*} \sum_{n \in S} q_n, \quad \forall S \subseteq S_{tot}. \tag{B.1}$$

In terms of necessary conditions, (B.1) certainly implies (3.21). Now, we prove the sufficient part by contradiction. Suppose that the whole system is not stabilizable. Therefore, there exists at least one smallest unstabilizable subset, say S^*. Let m be the largest element in S^*. If $S_m = S^*$, then the proof is complete. Otherwise, suppose that u is the largest element in $S_m \setminus S^*$ and $u < m$. Also, we define $S^{**} := S^* \cup \{u\}$. First, we want to show that

$$\frac{1}{r^*} \sum_{j \in S^{**}} q_j + \prod_{j \in S^{**}} (1 - p_j) > \frac{1}{r^*} \sum_{j \in S^*} q_j + \prod_{j \in S^*} (1 - p_j). \tag{B.2}$$

By subtracting the right-hand side of (B.2) from the left-hand side of (B.2), we just need to prove that

$$\frac{q_u}{r^*} - p_u \prod_{j \in S^*} (1 - p_j) = p_u \left(\frac{q_u}{p_u \cdot r^*} - \prod_{j \in S^*} (1 - p_j) \right) > 0 \tag{B.3}$$

Since S^* is a smallest unstabilizable set, we have

$$1 - \prod_{n \in S^*} (1 - p_n) < \frac{1}{r^*} \sum_{n \in S^*} q_n \tag{B.4}$$

By choosing $\tilde{S} = S^* \setminus \{m\}$ in (B.4), $\tilde{S}$ should be stabilizable and hence we have

$$1 - \prod_{n \in \tilde{S}} (1 - p_n) \geq \frac{1}{r^*} \sum_{n \in \tilde{S}} q_n. \tag{B.5}$$

From (B.4) and (B.5), we have

$$\frac{1}{r^*} \sum_{j \in S^*} q_j + \prod_{j \in S^*} (1 - p_j) > \frac{1}{r^*} \sum_{j \in \tilde{S}} q_j + \prod_{j \in \tilde{S}} (1 - p_j) \tag{B.6}$$

or equivalently, $\frac{q_m}{p_m \cdot r^*} > \prod_{j \in \tilde{S}} (1 - p_j)$. Since $u < m$, we can obtain that

$$\frac{q_u}{p_u \cdot r^*} \geq \frac{q_m}{p_m \cdot r^*} > \prod_{j \in \tilde{S}} (1 - p_j) > \prod_{j \in S^*} (1 - p_j). \tag{B.7}$$

Hence, both (B.2) and (B.3) hold. If $S_m = S^{**}$, the proof is complete. Otherwise, we continue by finding the largest element in $S_m \setminus S^{**}$ and repeat the same procedure shown in (B.2)–(B.7). By induction, finally we have

$$\frac{1}{r^*} \sum_{j \in S_m} q_j + \prod_{j \in S_m} (1 - p_j) > \frac{1}{r^*} \sum_{j \in S^*} q_j + \prod_{j \in S^*} (1 - p_j) > 1,$$

which contradicts the condition given by (3.21). Since the time complexity of this algorithm is dominated by the pre-sorting of $\frac{q_n}{p_n}$, the overall complexity is $O(N \log N)$. □

B.2 Proof of Theorem 12

We prove the state space collapse property by introducing a fluid system. First, define

$$V_n(t) = -w_n X_n(t) + \frac{\sum_{m=1}^N X_m(t)}{\sum_{m=1}^N \frac{1}{w_m}}. \tag{B.8}$$

Then, the largest $V_n(t)$ is associated with the client with the smallest $w_n X_n(t)$. By noting that $\frac{\sum_{m=1}^N X_m(t)}{\sum_{m=1}^N \frac{1}{w_m}}$ is a weighted average of $w_m X_m(t)$, we also have $\max_{1\le m\le N} V_m \ge 0$ and the equality holds if and only if $w_n X_n(t) = w_m X_m(t)$, for any pair n, m in S_{tot}. Next, we consider the *fluid limit* of $V_n(t)$ defined as

$$\bar{V}_n(t) := \lim_{k\to\infty} \frac{V_n(kt)}{k} = -w_n \bar{X}_n(t) + \frac{\sum_{m=1}^N \bar{X}_m(t)}{\sum_{m=1}^N \frac{1}{w_m}}, \tag{B.9}$$

where $\bar{X}_n(t) := \lim_{k\to\infty} \frac{X_n(kt)}{k}$ is the fluid limit for $X_n(t)$. Define a Lyapunov function

$$L(t) = \sum_{n=1}^N \frac{1}{2w_n} [\bar{V}_n(t)]^2. \tag{B.10}$$

Without loss of generality, we may assume that fluid limits of $V_m(t)$ are sorted in descending order, i.e. $\bar{V}_1(t) \ge \bar{V}_2(t) \ge \cdots \ge \bar{V}_N(t)$. Now, we derive the *Lyapunov drift* as

$$\frac{dL(t)}{dt} = \sum_{n=1}^N \frac{1}{w_n} \bar{V}_n(t) \frac{d\bar{V}_n(t)}{dt} \tag{B.11}$$

where

$$\frac{d\bar{V}_n(t)}{dt} = -w_n \frac{d\bar{X}_n(t)}{dt} + \frac{\sum_{m=1}^N \frac{d\bar{X}_m(t)}{dt}}{\sum_{m=1}^N \frac{1}{w_m}}. \tag{B.12}$$

Under JCD policy, client n is scheduled when $r_n > 0$ and $r_m = 0$, for all $m < n$. Let $\tilde{p}_n := Pr[(\bigcap_{k=1}^{n-1} W_k^c) \cap W_n]$. Since $X_n(t) = A_n(t) - q_n t$, we have

$$\frac{d\bar{X}_n(t)}{dt} = r^* \tilde{p}_n - q_n, \tag{B.13}$$

We further define

$$g_n := \sum_{k=1}^{n} \frac{d\bar{X}_k(t)}{dt} = r^* \sum_{k=1}^{n} \tilde{p}_k - \sum_{k=1}^{n} q_k \tag{B.14}$$

By using the conditions given by (3.22) and (3.23), we have

$$\begin{cases} g_k > 0, & \text{if } k = 1, 2, ..., N-1 \\ g_k = 0, & \text{if } k = N \end{cases} \tag{B.15}$$

For convenience, we also let $g_0 = 0$. Thus, we can rewrite (B.12) as $\frac{d\bar{V}_n(t)}{dt} = -w_n(g_n - g_{n-1})$. Finally, the Lyapunov drift in (B.11) can be computed as

$$\begin{aligned} \frac{dL(t)}{dt} &= -\sum_{n=1}^{N} (g_n - g_{n-1}) \cdot \bar{V}_n(t) \\ &= -\left[\left(\sum_{n=1}^{N-1} g_n \cdot (\bar{V}_n(t) - \bar{V}_{n+1}(t)) \right) + g_N \bar{V}_N(t) \right] \leq 0 \end{aligned}$$

Moreover, the drift is zero only if $\bar{V}_1(t) = \bar{V}_2(t) = \cdots = \bar{V}_N(t) = 0$. Therefore, the random process $\{V_n(t)\}$ is *positive recurrent*. Thus, we have

$$\hat{V}_n(t) := \lim_{k \to \infty} \frac{V_n(kt)}{\sqrt{k}} = -w_n \hat{X}_n(t) + \frac{\sum_{m=1}^{N} \hat{X}_m(t)}{\sum_{m=1}^{N} \frac{1}{w_m}} = 0.$$

This also implies that $w_n\hat{X}_n(t) = w_m\hat{X}_m(t)$ for any pair of n, m and thus completes the proof. □

B.3 Proof of Lemma 9

Proof. Define $\widetilde{C}_n(j) := C_n(j) - jq_n^*$ and $\Delta\widetilde{C}_n(j+1) := \widetilde{C}_n(j+1) - \widetilde{C}_n(j)$. By the i.i.d. assumption on frame size, $\Delta\widetilde{C}_n(j)$ is i.i.d. across all time slots. Moreover,

$$E[\Delta\widetilde{C}_n(j)] = E[F_n(j) - q_n^*] = E[F_n(j)] - q_n^* = 0,$$
$$\mathrm{Var}[\Delta\widetilde{C}_n(t)] = \mathrm{Var}[F_n(j) - q_n^*] = \mathrm{Var}[F_n(j)] = \sigma_{q,n}^2.$$

By the functional central limit theorem for i.i.d. random variables, we know $\hat{C}_n(t)$ is a driftless Brownian motion with variance $\sigma_{q,n}^2$. Hence, $\hat{C}_n(\frac{t}{k_n})$ is a driftless Brownian motion with variance $\sigma_{q,n}^2/k_n$. Next, consider $\hat{Z}_n(t)$ as

$$\hat{Z}_n(t) = \lim_{k\to\infty} \frac{C_n(\lfloor\frac{kt}{k_n}\rfloor) - q_n kt}{\sqrt{k}} \tag{B.16}$$
$$= \lim_{k\to\infty} \frac{C_n(k\frac{\lfloor\frac{kt}{k_n}\rfloor}{k}) - q_n^* k\frac{t}{k_n}}{\sqrt{k}} = \hat{C}_n\Big(\frac{t}{k_n}\Big). \tag{B.17}$$

Finally, we consider $\hat{Y}_n(t)$ as

$$\hat{Y}_n(t) = \lim_{k\to\infty} \frac{C_n(S_n(kt)) - q_n(kt - D_n(kt)) + e_n(kt)}{\sqrt{k}} \tag{B.18}$$
$$= \lim_{k\to\infty} \frac{C_n(\lfloor\frac{kt-D_n(kt)}{k_n}\rfloor) - q_n(kt - D_n(kt)) + e_n(kt)}{\sqrt{k}} \tag{B.19}$$
$$= \lim_{k\to\infty} \frac{C_n(k\frac{\lfloor\frac{kt-D_n(kt)}{k_n}\rfloor}{k}) - q_n(k(t - \frac{D_n(kt)}{k})) + e_n(kt)}{\sqrt{k}} \tag{B.20}$$
$$= \hat{C}_n\Big(\frac{t}{k_n}\Big). \tag{B.21}$$

The above is true according to the Random Time-Change Theorem (Theorem 5.3 in [19]). In summary, we have $\hat{Y}_n(t) = \hat{Z}_n(t) = \hat{C}_n\left(\frac{t}{k_n}\right)$. □ □

B.4 Proof of Theorem 17

We prove the state space collapse property by introducing a fluid system. First, define

$$Q_n(t) = -w_n(X_n(t) - Z_n(t)) + \frac{\sum_{m=1}^{N}(X_m(t) - Z_m(t))}{\sum_{m=1}^{N}\frac{1}{w_m}}. \tag{B.22}$$

Two important facts of the above definition:

- At any time t, the client n with the largest $Q_n(t)$ also has the largest $-w_n(X_n(t) - Z_n(t))$.
- Since $\frac{\sum_{m=1}^{N}(X_m(t)-Z_m(t))}{\sum_{m=1}^{N}\frac{1}{w_m}}$ is the weighted average of $w_n(X_n(t) - Z_n(t))$, we have $\max_{1\le m\le N} Q_m \ge 0$.

Next, we study the fluid limit of $Q_n(t)$ defined as

$$\bar{Q}_n(t) := \lim_{k\to\infty} \frac{Q_n(kt)}{k} \tag{B.23}$$

Similarly, define $\bar{X}_n(t) := \lim_{k\to\infty} \frac{X_n(kt)}{k}$ and $\bar{Z}_n(t) := \lim_{k\to\infty} \frac{Z_n(kt)}{k}$ to be the fluid limits of $X_n(t)$ and $Z_n(t)$, respectively. Since $Z_n(t) := C_n(\lfloor \frac{t}{k_n} \rfloor) - q_n t$, we thus have

$$\bar{Z}_n(t) = \lim_{k\to\infty} \frac{C_n(\lfloor \frac{kt}{k_n} \rfloor) - q_n kt}{k} = \lim_{k\to\infty} \frac{C_n(k\frac{\lfloor \frac{kt}{k_n} \rfloor}{k})}{k} - q_n t \tag{B.24}$$

Since $\lim_{k\to\infty} \frac{\lfloor \frac{kt}{k_n} \rfloor}{k} = \frac{t}{k_n}$, then by the Random Time-Change Theorem (Theorem 5.3 in [19]), we have $\lim_{k\to\infty} \frac{C_n(k\frac{\lfloor \frac{kt}{k_n} \rfloor}{k})}{k} = q_n^* \frac{t}{k_n} = q_n t$. Hence, $\bar{Z}_n(t) = 0$, for any $t \ge 0$,

for every client n. Thus, $\bar{Q}_n(t)$ can be written as

$$\bar{Q}_n(t) = -w_n(\bar{X}_n(t) - \bar{Z}_n(t)) + \frac{\sum_{m=1}^{N}(\bar{X}_m(t) - \bar{Z}_m(t))}{\sum_{m=1}^{N}\frac{1}{w_m}} \tag{B.25}$$

$$= -w_n\bar{X}_n(t) + \frac{\sum_{m=1}^{N}\bar{X}_m(t)}{\sum_{m=1}^{N}\frac{1}{w_m}}. \tag{B.26}$$

The rest of the proof is to show that the random process $\{Q_n(t)\}$ is positive recurrent for all n. Define a Lyapunov function

$$L_Q(t) = \sum_{n=1}^{N}\frac{1}{2w_n}[\bar{Q}_n(t)]^2. \tag{B.27}$$

We again assume that fluid limits of $Q_m(t)$ are sorted in descending order, i.e. $\bar{Q}_1(t) \geq \bar{Q}_2(t) \geq \cdots \geq \bar{Q}_N(t)$. Let U_n be the event that $r_n(t)$ equals $R(t)$ at some given time t. Since $X_n(t) = A_n(t) - q_n t$, under the HDR-VBR policy we have

$$\frac{d\bar{X}_n(t)}{dt} = E\left[R(t)\cdot\mathbb{I}\left\{\left(\bigcap_{k=1}^{n-1}U_k^c\right)\cap U_n\right\}\right] - q_n, \tag{B.28}$$

where $\{\left(\bigcap_{k=1}^{n-1}U_k^c\right)\cap U_n\}$ represents the event that client n is the only client in $\{1,2,...,n\}$ which has the largest transmission rate among all clients. Now, let $\tilde{r}_n := E\left[R(t)\cdot\mathbb{I}\left\{\left(\bigcap_{k=1}^{n-1}U_k^c\right)\cap U_n\right\}\right]$. Then, we define $h_k := \sum_{j=1}^{k}(\tilde{r}_j - q_j) = E\left[R(t)\cdot\mathbb{I}\left\{\bigcup_{j=1}^{k}U_j\right\}\right] - \sum_{j=1}^{k}q_j$, where $\{\bigcup_{j=1}^{k}U_j\}$ represents the event that at least one client in $\{1,2,..,k\}$ has the largest transmission rate among all clients. By using the conditions in (3.37) and (3.38), we obtain that

$$\begin{cases} h_k > 0, & \text{if } k = 1,2,...,N-1 \\ h_k = 0, & \text{if } k = N \end{cases} \tag{B.29}$$

where the last equality holds since $\sum_{m=1}^{N}(\tilde{r}_m - q_m) = h_N$ and should be zero. For convenience, we also let $h_0 = 0$. Thus, we have

$$\frac{d\bar{Q}_n(t)}{dt} = -w_n(\tilde{r}_n - q_n) + \frac{\sum_{m=1}^{N}(\tilde{r}_m - q_m)}{\sum_{m=1}^{N}\frac{1}{w_m}} = -w_n(\tilde{r}_n - q_n), \tag{B.30}$$

Finally, the Lyapunov drift is given by

$$\begin{aligned}
\frac{dL_Q(t)}{dt} &= -\sum_{n=1}^{N}(\tilde{r}_n - q_n)\cdot\bar{Q}_n(t) \\
&= -\sum_{n=1}^{N}(h_n - h_{n-1})\cdot\bar{Q}_n(t) \\
&= -\left[\left(\sum_{n=1}^{N-1} h_n\cdot\left(\bar{Q}_n(t) - \bar{Q}_{n+1}(t)\right)\right) + h_N\bar{Q}_N(t)\right] \\
&\leq 0.
\end{aligned}$$

Note that the drift is zero only if $\bar{Q}_1(t) = \bar{Q}_2(t) = \cdots = \bar{Q}_N(t) = 0$. Hence, the random process $\{Q_n(t)\}$ is positive recurrent. Therefore, $\hat{Q}_n(t) = 0$, for every n. This result implies that $w_n(\hat{X}_n(t) - \hat{Z}_n(t)) = w_m(\hat{X}_m(t) - \hat{Z}_m(t))$, for every pair n, m. $\square$

APPENDIX C

PROOFS OF CHAPTER 4

C.1 Proof of Lemma 11

Proof. By the assumption that the influence function $f(\cdot)$ is Riemann integrable, define $V(x) := \int_0^x f(t)dt$. We define a Lyapunov function as

$$L(k) := \sum_{n=1}^{N} V(d_n^+(k)), \forall k \geq 0. \tag{C.1}$$

Let $\Delta d_n(k) := d_n^+(k+1) - d_n^+(k)$. Consider the Lyapunov drift $\Delta(L(k)) := \mathbb{E}\big[L(k+1) - L(k) \,\big|\, \boldsymbol{d}(k)\big]$. Then, we have

$$\Delta(L(k)) \tag{C.2}$$

$$= \mathbb{E}\left[\sum_{n=1}^{N} V(d_n^+(k+1)) - V(d_n^+(k)) \,\middle|\, \boldsymbol{d}(k)\right] \tag{C.3}$$

$$= \mathbb{E}\left[\sum_{n=1}^{N} \frac{V(d_n^+(k+1)) - V(d_n^+(k))}{\Delta d_n(k)} \cdot \Delta d_n(k) \,\middle|\, \boldsymbol{d}(k)\right]. \tag{C.4}$$

Note that in (C.4) we implicitly assume that $\Delta d_n(k) \neq 0$ since $V(d_n^+(k+1)) - V(d_n^+(k))$ is nonzero only when $\Delta d_n(k) \neq 0$. Let $m_n(k) = \min\{d_n^+(k), d_n^+(k+1)\}$ and $M_n(k) = \max\{d_n^+(k), d_n^+(k+1)$, for each n and for any $k \geq 0$. Since $V(\cdot)$ is differentiable, then by the Mean Value Theorem, we know that there exists a real number $c_n(k) \in [m_n(k), M_n(k)]$ such that $V'(c_n(k)) = \frac{V(d_n^+(k+1)) - V(d_n^+(k))}{\Delta d_n(k)}$. For any $t \geq 0$, by the evolution of d_n specified by (4.1), we have $|d_n(t+1) - d_n(t)| \leq \max\{q_{\max}, A_{\max}\}$, where $q_{\max} := \max_{1 \leq n \leq N} q_n$. Moreover, by (4.1), we know $\Delta d_n(k) = q_n - S_n(k)$ if $d_n(k) \geq A_{\max}$, for all n and for all $k \geq 0$. Here we can focus on the case where

$d_n(k) \geq A_{\max}$ for all n since $V^{'}(c_n(k)) = f(c_n(k))$ is upper bounded by some constant if $d_n(k) < A_{\max}$. Therefore, we can rewrite (C.4) as

$$\Delta(L(k)) = \mathbb{E}\Big[\sum_{n=1}^{N} V^{'}(c_n(k)) \cdot (q_n - S_n(k)) \,\Big|\, \boldsymbol{d}(k)\Big] \tag{C.5}$$

$$= \mathbb{E}\Big[\sum_{n=1}^{N} f(c_n(k)) \cdot (q_n - S_n(k)) \,\Big|\, \boldsymbol{d}(k)\Big]. \tag{C.6}$$

Suppose $\boldsymbol{q}$ is strictly feasible, then there exists some $\alpha \in (0,1)$ such that $(1+\alpha)\boldsymbol{q}$ is also feasible. Since the packet arrivals are i.i.d. across intervals, then there exists a stationary randomized policy $\eta^{'}$ such that $\mathbb{E}^{\eta^{'}}[S_n(k)] \geq q_n(1+\alpha),\ \forall n \in \mathcal{N}, \forall k \geq 0$. Since $|d_n(t+1) - d_n(t)| \leq \max\{q_{\max}, A_{\max}\}$, then by the property of $f(\cdot)$ specified in Section 4.2, given any $\epsilon > 0$ there exists some $D_n > 0$ such that

$$1 - \epsilon \leq \frac{f(c_n(k))}{f(d_n^+(k))} \leq 1 + \epsilon, \tag{C.7}$$

whenever $d_n(k) \geq D_n$. Meanwhile, if $d_n(k) < D_n$, we know $f(c_n(k))$ is upper bounded by some positive constant B_n. Then, we can rewrite (C.6) as

$$\begin{aligned}
&\Delta(L(k)) \\
&\leq \mathbb{E}\Big[\sum_{n=1}^{N} f(d_n^+(k)) \cdot ((1+\epsilon)q_n - (1-\epsilon)S_n(k)) \,\Big|\, \boldsymbol{d}(k)\Big] + B \\
&\leq \mathbb{E}^{\eta^{'}}\Big[\sum_{n=1}^{N} f(d_n^+(k)) \cdot ((1+\epsilon)q_n \,\Big|\, \boldsymbol{d}(k)\Big] \\
&\qquad - (1-\epsilon)(1-\delta)\,\mathbb{E}^{\eta^{'}}\Big[\sum_{n=1}^{N} f(d_n^+(k))S_n(k)) \,\Big|\, \boldsymbol{d}(k)\Big] + B \\
&\leq \sum_{n=1}^{N} f(d_n^+(k)) \cdot \Big((1+\epsilon)q_n - (1-\epsilon)(1-\delta)(1+\alpha)q_n\Big) + B,
\end{aligned}$$

where $B := \sum_{n=1}^{N} B_n q_n < \infty$. By choosing sufficiently small ϵ and δ, for example, $\epsilon = \frac{\alpha}{8}$ and $\delta = \frac{\alpha}{8}$, we have

$$\Delta(L(k)) \leq B + \sum_{n=1}^{N} f(d_n^+(k)) \cdot \left((1+\frac{\alpha}{8})q_n - (1+\frac{\alpha}{2})q_n\right) \tag{C.8}$$

$$\leq B - \frac{3\alpha q_{\min}}{8} \sum_{n=1}^{N} f(d_n^+(k)), \tag{C.9}$$

where $q_{\min} := \min_{n\in\mathcal{N}} q_n$. By Foster's Criterion [128], we know that the Markov chain induced by $\{[f(d_n^+(k)), n \in \mathcal{N}]\}$ is positive recurrent. This implies that there exists a unique stationary distribution for the Markov chain induced by $\{[f(d_n^+(k)), n \in \mathcal{N}]\}$. Due to monotonicity of $f(\cdot)$ described in Definition 6, the Markov chain induced by $\{\boldsymbol{d}(k)\}$ is also positive recurrent. This implies that $\frac{d_n^+(k)}{k} \to 0$ in probability as $k \to \infty$, for all n. Therefore, we conclude that η is feasibility-optimal. □

C.2 Proof of Lemma 13

Proof. First, we show that the Markov chain induced by $\{\boldsymbol{\sigma}(k)\}$ is irreducible. It is evident that starting from any permutation in $\mathfrak{S}_N$, any other permutation in $\mathfrak{S}_N$ can be achieved by repeatedly applying adjacent transpositions to the original sequence. As shown in Algorithm 3, in each interval, any pair of links i, j that forms an adjacent transposition is chosen with probability $\frac{1}{N-1}$ as candidates for swapping priority indices under the proposed algorithm. Under condition (C1), the candidate links i, j shall exchange their priority indices with nonzero probability as shown in (4.9). Therefore, given any $\boldsymbol{\sigma}(k) \in \mathfrak{S}_N$ in interval k, the Markov chain can move to any other state in finite steps with nonzero probability. Hence, the Markov chain $\{\boldsymbol{\sigma}(k)\}$ is indeed irreducible.

To show that $\{\boldsymbol{\sigma}(k)\}$ is aperiodic, we only need to show that there exists one aperiodic state since $\{\boldsymbol{\sigma}(k)\}$ is irreducible. Pick any arbitrary $\boldsymbol{\sigma} \in \mathfrak{S}_N$. The recur-

rence time of $\boldsymbol{\sigma}$ can be two or three time intervals with nonzero probability under Algorithm 3. Therefore, $\boldsymbol{\sigma}$ is aperiodic and hence the Markov chain $\{\boldsymbol{\sigma}(k)\}$ is also aperiodic. □

C.3 Proof of Proposition 4

Proof. Without loss of generality, suppose $f(d_1^+(k))p_1 \geq f(d_2^+(k))p_2 \geq \cdots \geq f(d_N^+(k))p_N$. By the ELDF policy, we know that the optimal transmission priority vector for the k-th interval is $\boldsymbol{\sigma}^*(k) = (1, 2, ..., N)$. By the timescale-separation argument, we suppose that the Markov Chain induced by the debt vector $\boldsymbol{d}(k)$ is already in steady state in the k-th interval. The key concept of this proof is to show that under the priority-based policy, $\mathbb{E}^{\eta}\left[\sum_{n=1}^{N} f(d_n^+(k))S_n(k) \,\middle|\, \boldsymbol{d}(k)\right]$ is close to maximum with high probability. We first introduce the concept of *inversion* as follows.

Definition 11. *Suppose $f(d_1^+(k))p_1 \geq f(d_2^+(k))p_2 \geq \cdots \geq f(d_N^+(k))p_N$. A transmission priority vector $\boldsymbol{\sigma}$ is said to have an* inversion *of parameter α_0 if there exists a pair of links $i, j \in \{1, ..., N\}$ such that $\sigma_i > \sigma_j$ and $f(d_i^+(k))p_i \geq f(d_j^+(k))p_j + \alpha_0$.*

Let $\alpha^* = \epsilon_0 f(d_1^+(k))p_1$, where ϵ_0 is a small positive number. Based on the concept of inversion, we can divide $\mathfrak{S}_N$ into two disjoint subsets. Let $\mathcal{L}$ be the set of priority vectors $\boldsymbol{\sigma} \in \mathfrak{S}_N$ with no inversion of parameter α_0. For each priority vector $\boldsymbol{\sigma} \in \mathfrak{S}_N$, $\boldsymbol{\sigma}$ shall be in one of the following cases:

- Case 1: $\boldsymbol{\sigma} \in \mathcal{L}$, i.e. $\boldsymbol{\sigma}$ has no inversion of parameter α_0.
- Case 2: $\boldsymbol{\sigma} \in \mathfrak{S}_N \setminus \mathcal{L}$, i.e. $\boldsymbol{\sigma}$ has at least one inversion of parameter α_0.

We start from Case 1. For any transmission priority vector $\boldsymbol{\sigma} \in \mathcal{L}$, the following lemma shows that $\boldsymbol{\sigma}$ achieves close to maximum $\mathbb{E}\left[\sum_{n=1}^{N} f(d_n^+(k))S_n(k) \,\middle|\, \boldsymbol{d}(k)\right]$.

Lemma 16. *For any $k \geq 0$ and for any transmission priority vector $\boldsymbol{\sigma} \in \mathcal{L}$, we have*

$$\mathbb{E}^{\boldsymbol{\sigma}}\left[\sum_{n=1}^{N} f(d_n^+(k))S_n(k) \,\middle|\, \boldsymbol{d}(k)\right] \tag{C.10}$$

$$\geq \mathbb{E}^{\boldsymbol{\sigma}^*}\left[\sum_{n=1}^{N} f(d_n^+(k))S_n(k) \,\middle|\, \boldsymbol{d}(k)\right] - 2^{T-1}\alpha_0. \tag{C.11}$$

Proof. (Lemma 16) Consider $\boldsymbol{U}(t) = (U_1(t), \cdots, U_N(t))$, where $U_n(t)$ denotes the number of undelivered packets of link n at the t-th time slot of an interval. Let $\hat{S}_n(\boldsymbol{U}(t), t)$ be the random variable of the number of delivered packets of link n between the t-th slot and the T-th slot of an interval given the undelivered packets $\boldsymbol{U}(t)$. Let $W^{\boldsymbol{\sigma}}(\boldsymbol{U}(t), t)$ be the value of $\mathbb{E}^{\boldsymbol{\sigma}}\left[\sum_{n=1}^{N} f(d_n^+(k))\hat{S}_n(\boldsymbol{U}(t), t) \,\middle|\, \boldsymbol{d}(k)\right]$ under the transmission priority vector $\boldsymbol{\sigma}$. We claim that for any $\boldsymbol{\sigma} \in \mathcal{L}$ and given any $t \in \{1, \cdots, T\}$ and any $\boldsymbol{U}(t)$,

$$W^{\boldsymbol{\sigma}}(\boldsymbol{U}(t), t) \geq W^{\boldsymbol{\sigma}^*}(\boldsymbol{U}(t), t) - 2^{T-t}\alpha_0. \tag{C.12}$$

We prove the above claim by induction.

(i) When $t = T$, there is exactly one transmission to be made. Suppose under the transmission priority vector $\boldsymbol{\sigma}$ and $\boldsymbol{\sigma}^*$, this transmission at $t = T$ is done by some

link i and j, respectively. Since $\boldsymbol{\sigma} \in \mathcal{L}$, then by the definition of $\mathcal{L}$ we have

$$W^{\boldsymbol{\sigma}}(\boldsymbol{U}(T), T) = f(d_i^+(k))p_i \tag{C.13}$$

$$\geq f(d_j^+(k))p_j - \alpha_0 \tag{C.14}$$

$$= W^{\boldsymbol{\sigma}^*}(\boldsymbol{U}(T), T) - \alpha_0. \tag{C.15}$$

(ii) Next, assume that the claim of (C.12) holds for $t = \tau + 1, \cdots, T$. Suppose at time $t = \tau$, given $\boldsymbol{U}(\tau)$, the transmission is made by some links i and j under transmission priority $\boldsymbol{\sigma}$ and $\boldsymbol{\sigma}^*$, respectively. We use $\boldsymbol{e}_i$ to denote an N-dimensional unit vector with i-th entry equal to 1. If $i = j$, then we have

$$W^{\boldsymbol{\sigma}}(\boldsymbol{U}(\tau), \tau) \tag{C.16}$$

$$= f(d_i^+(k))p_i + p_i W^{\boldsymbol{\sigma}}(\boldsymbol{U}(\tau) - \boldsymbol{e}_i, \tau + 1) \tag{C.17}$$

$$+ (1 - p_i)W^{\boldsymbol{\sigma}}(\boldsymbol{U}(\tau), \tau + 1) \tag{C.18}$$

$$\geq f(d_i^+(k))p_i + p_i\Big[W^{\boldsymbol{\sigma}^*}(\boldsymbol{U}(\tau) - \boldsymbol{e}_i, \tau + 1) - 2^{T-\tau-1}\alpha_0\Big] \tag{C.19}$$

$$+ (1 - p_i)\Big[W^{\boldsymbol{\sigma}^*}(\boldsymbol{U}(\tau), \tau + 1) - 2^{T-\tau-1}\alpha_0\Big] \tag{C.20}$$

$$= W^{\boldsymbol{\sigma}^*}(\boldsymbol{U}(\tau), \tau) - 2^{T-\tau-1}\alpha_0. \tag{C.21}$$

Otherwise, if $i \neq j$, we can consider another policy π' which follows $\boldsymbol{\sigma}$ at time τ and follows $\boldsymbol{\sigma}^*$ from $\tau + 1$ to T. Under policy π', since link j maximizes $f(d_j^+(k))p_j$, we know that transmission in $(\tau + 1)$-th time slot is made by link j regardless of the result of transmission made by link i in the τ-th slot. Accordingly, the policy π' is equivalent to another policy π'' that schedules link j in τ-th slot, schedules link i in $(\tau + 1)$-th slot, and follows $\boldsymbol{\sigma}^*$ from $\tau + 2$ to T. Note that the resulting transmission

priority vectors under $\pi^{'}$ and $\pi^{''}$ are also in the set $\mathcal{L}$. Therefore, we have

$$W^{\boldsymbol{\sigma}}(\boldsymbol{U}(\tau),\tau) \geq W^{\pi^{'}}(\boldsymbol{U}(\tau),\tau) - 2^{T-\tau-1}\alpha_0 \tag{C.22}$$

$$= W^{\pi^{''}}(\boldsymbol{U}(\tau),\tau) - 2^{T-\tau-1}\alpha_0 \tag{C.23}$$

$$\geq \Big(W^{\boldsymbol{\sigma}^*}(\boldsymbol{U}(\tau),\tau) - 2^{T-\tau-1}\alpha_0\Big) - 2^{T-\tau-1}\alpha_0, \tag{C.24}$$

where the last inequality in (C.24) follows directly from (C.16)-(C.21). We complete the proof of the claim of (C.12). Then, (C.10) and (C.11) follow directly from (C.12) when $t = 1$. This completes the proof of Lemma 16. □

Now we are ready to finish the rest of the proof of Proposition 4. For Case 2 where $\boldsymbol{\sigma} \in \mathfrak{S}_N \setminus \mathcal{L}$, by the stationary distribution π^* derived in Proposition 3, we have

$$\frac{\pi^*(\boldsymbol{\sigma}^*)}{\pi^*(\boldsymbol{\sigma}^*)} \geq \exp(\alpha_0) = \exp\big(\epsilon_0 f(d_1^+(k))p_1\big). \tag{C.25}$$

Therefore, in steady state, we have

$$\Pr(\boldsymbol{\sigma}^{'} \in \mathcal{L}) \geq \frac{\exp\big(\epsilon_0 f(d_1^+(k))p_1\big)}{\exp\big(\epsilon_0 f(d_1^+(k))p_1\big) + N!} \tag{C.26}$$

$$= 1 - \frac{N!}{\exp\big(\epsilon_0 f(d_1^+(k))p_1\big) + N!}. \tag{C.27}$$

By choosing $\epsilon_0 = \frac{\delta}{2^T}$, we have

$$\mathbb{E}^{\eta}\left[\sum_{n=1}^{N} f(d_n^+(k))S_n(k) \,\middle|\, \boldsymbol{d}(k)\right] \tag{C.28}$$

$$\geq \Pr(\boldsymbol{\sigma}' \in \mathcal{L})\left(\mathbb{E}^{\boldsymbol{\sigma}^*}\left[\sum_{n=1}^{N} f(d_n^+(k))S_n(k) \,\middle|\, \boldsymbol{d}(k)\right] - 2^{T-1}\alpha_0\right) \tag{C.29}$$

$$\geq \Pr(\boldsymbol{\sigma}' \in \mathcal{L}) \cdot \left(1 - 2^{T-1}\epsilon_0\right) \tag{C.30}$$

$$\cdot\, \mathbb{E}^{\boldsymbol{\sigma}^*}\left[\sum_{n=1}^{N} f(d_n^+(k))S_n(k) \,\middle|\, \boldsymbol{d}(k)\right] \tag{C.31}$$

$$= \left(1 - \frac{N!}{\exp\big(\epsilon_0 f(d_1^+(k))p_1\big) + N!}\right) \cdot (1 - \frac{\delta}{2}) \tag{C.32}$$

$$\cdot\, \mathbb{E}^{\boldsymbol{\sigma}^*}\left[\sum_{n=1}^{N} f(d_n^+(k))S_n(k) \,\middle|\, \boldsymbol{d}(k)\right] \tag{C.33}$$

Let $R^* = \frac{1}{\epsilon_0}\ln(\frac{2N!}{\delta} - N!)$. When $f(d_1^+(k))p_1 \geq R^*$, we have

$$\mathbb{E}^{\eta}\left[\sum_{n=1}^{N} f(d_n^+(k))S_n(k) \,\middle|\, \boldsymbol{d}(k)\right] \tag{C.34}$$

$$\geq (1 - \frac{\delta}{2}) \cdot (1 - \frac{\delta}{2}) \cdot \mathbb{E}^{\boldsymbol{\sigma}^*}\left[\sum_{n=1}^{N} f(d_n^+(k))S_n(k) \,\middle|\, \boldsymbol{d}(k)\right] \tag{C.35}$$

$$\geq (1 - \delta) \cdot \max_{\eta' \in \Pi} \mathbb{E}^{\eta'}\left[\sum_{n=1}^{N} f(d_n^+(k))S_n(k) \,\middle|\, \boldsymbol{d}(k)\right], \tag{C.36}$$

where the last inequality in (C.36) follows from the fact that the ELDF policy maximizes $\mathbb{E}\left[\sum_{n=1}^{N} f(d_n^+(k))S_n(k) \,\middle|\, \boldsymbol{d}(k)\right]$ in every interval. Define the inverse function of $f(\cdot)$ as $R^{-1}(r) := \inf\{x \in \mathbb{R}_{\geq 0} : f(x) \geq r\}$ and $p_{\min} := \min_n p_n$. Choose $B = \max_n R^{-1}(\frac{R^*}{p_{\min}})$. Then, we know $|\boldsymbol{d}(k)|_\infty > B$ implies that $f(d_1^+(k))p_1 \geq R^*$. This completes the proof. □